Universitat Oberta de Catalunya

Proyecto Final de Carrera

Mecano Virtual 3D

Alumno: Fco. Javier Bujalance López

Consultor: Pere Pau Vázquez Alcocer

UOC
ETIS. Informática Gráfica
Enero 2005

Lista de cambios

Número	Fecha	Descripción	Autor / es
1	13-Dic-04	Versión 1.0 Primera versión para revisión conjunta con consultor	F.J. Bujalance L.
2	10-ene-05	Versión 2.0 Versión final añadidos anexos	F.J. Bujalance L.

Resumen

Hoy en día, la mayoría de la aplicaciones software dedican muchos recursos a la interacción con él usuario. Se puede decir que uno de los objetivos comunes de los programas que se realizan actualmente es que su uso sea muy intuitivo.

Es bastante evidente que si reproducimos gráficamente el entorno en el que el usuario se movería usando la aplicación software minimizamos la adaptación de dicho usuario al software.

La informática gráfica pretende conseguir este objetivo y para ello dota de los conocimientos teóricos y de las herramientas para realizarlo. Una de estas herramientas es el OpenGL o su versión para Java, GL4Java.

Por otra parte, los procesos productivos actuales, en vistas a ser más competitivos, minimizan los plazos desde el desarrollo de un producto a su puesta en fabricación. Para conseguir esto, se dota a los diseñadores del producto y de los útiles de producción de unos simuladores para poder reproducir la Pieza, el Conjunto de piezas, incluso la Máquina antes de haber dado por terminado el proyecto.

El presente proyecto final de carrera ha querido adentrarse dentro de estas facetas para poder valorar que tipo de información sería necesario, que tipo de estructuras de datos harían falta y cuales podrían ser las perspectivas de uso de las herramientas OpenGL para este fin.

A la hora de escoger un entorno donde desarrollar estos puntos se pensó que debía ser lo suficientemente sencillo como para obtener unos primeros resultados dentro del periodo de ejecución del propio proyecto final de carrera, y lo suficientemente genérico para poder desarrollar cosas más complejas. Así es como se decidió que el entorno fuera el de un Mecano.

El presente proyecto ha desarrollado un sistema con dos tipos de piezas base y con una serie de operaciones base.

Las piezas base son tales que a partir de ellas se pueden generar otras más complejas, que podríamos llamar submontajes. Las operaciones base son las de observación detallada de los montajes.

Este proyecto es solo una base a partir de la cual se puede ir aumentando la cantidad y complejidad de las piezas, así como diversos tipos de simulación, ampliando de esta manera, los conocimientos sobre las características de dichos tipos de proyectos.

Estos tipos de proyectos tienen un gran futuro debido a que nos darían a conocer como funcionan las cosas por si solas, intrínsecamente, y podemos usar esta información para generar interfaces de usuario más inteligentes que los meramente representativos de una realidad.

Lista de palabras clave

Desplazamiento. Es un tipo de estímulo que se puede aplicar a un Montaje. Consiste es desplazar el conjunto de Piezas que contiene un Montaje de forma paralela a cada uno de los ejes X, Y y Z.

Estímulo. Es la acción que se aplica a una Pieza de un Montaje para ver el efecto que produce en ella y en las otras Piezas del Montaje. Puede ser de varios tipos, principalmente angular o de giro y de desplazamiento.

Giro. Es un tipo de estímulo que se puede aplicar a un Montaje. Consiste en girar el conjunto de Piezas que contiene un Montaje utilizando como eje de giro un eje paralelo a cada uno de los ejes X, Y y Z.

GL4Java. Librería similar a OpenGL para trabajar en Java.

Java. Lenguaje de programación Orientado a Objetos que permite crear Applets visibles desde una página web o documento html.

Mecano. Dentro del ámbito de este proyecto, es un conjunto de Piezas de varios tipos que se pueden unir formando un Montaje. A dicho Montaje se le pueden aplicar estímulos para comprender mejor su arquitectura.

Montar. Acción de seleccionar Piezas sueltas del Mecano para construir con ellas un Montaje.

Montaje. Conjunto de Piezas unidas entre si, respetando las características de cada Pieza suelta. Es la unidad mínima a la que de le puede aplicar un estímulo.

Netscape. Visualizador de páginas web o documentos html.

OpenGL. Librería con herramientas para poder tratar la información gráfica.

Orientado a Objetos. Se dice que un lenguaje es orientado a objetos si cada una de las entidades de la información se constituye en una clase para gestionar sus propios atributos y los servicios que pueda prestar a otras entidades.

Pieza. Unidad física más simple que constituye un Mecano.

Selección. Acción de escoger una Pieza del conjunto de Piezas sueltas, o escoger una Pieza del conjunto de las de un Montaje.

Simulación. Acción de aplicar el estímulo deseado a una Pieza de un Montaje y observar como se distribuye por el conjunto de las otras Piezas.

Summary

Nowadays, most of the software applications dedicate many resources to the interaction with user. It is possible to say that one of the common objectives of the programs that are made at the moment is that their use will be very intuitive.

It is quite evident that if we reproduce graphically the environment in which the user would move using the software application we will minimise the adaptation of this user to the software.

Graphical computer science tries to obtain this objective and for it. For that, it equips with the theoretical knowledge and the tools to make it. One of these tools is the OpenGL and its version for Java, GL4Java.

On the other hand, the actual productive processes, in order to be more competitive, reduce the terms from the development of a product to their star-up in manufacture. In order to obtain this, the product and tooling designers use simulators to be able to reproduce the Piece, the Set of pieces, over all the Machine before of to give by finished the project.

The present project has wanted to enter inside these facets to be able to value that type of information would be necessary, what type of data structures are necessary and as they could be the perspective of use of the OpenGL tools for this objective.

At the time to select the environment where develop these points I thought that it will be enough simple to obtain first results during the period of execution of the own project , and the sufficiently generic to develop more complex things. For all of that I decided that the environment was a Meccano.

The present project has developed a system with two types of base pieces and some base operations.

The base pieces are such that from them we can generate other more complex ones, who we could call subassemblies. The base operations are those for detailed observation of the assemblies.

This project is only a base from which it is possible to be increasing to the amount and complexity of the pieces, as well as diverse types of simulation, extending in this way, the knowledge on the characteristics of these types of projects.

These types of projects have great future because they would present to us as the single things work in case, intrinsically, and we can use this information to generate user interfaces more intelligent than the merely representative ones of a reality.

Keywords

Movement , Stimulus, Turn, GL4Java, Java, Meccano, to Assemble, Assembly, NetScape, OpenGL, Object Oriented, Piece, Selection, Simulation

Indice:

Lista de figuras

Lista de Tablas

1. Introducción

Es bien notorio que estamos sumergidos en la sociedad y el tiempo de la Información. Todo tipo de información llega hasta nosotros.

Nuestros sentidos no paran de recibir información. Pero quizás sea la vista el sentido a partir del cual es más fácil transmitir información al cerebro, ya que casi no hay que tratar esta información para que quede retenida.

De hecho, la parte más importante del aprendizaje es visual. También podemos decir, sobre el aprendizaje, que hasta que una cosa no la hemos probado no nos quedamos con la esencia de su funcionamiento, incluso no creemos que funcione.

La unión de ambos aspectos del aprendizaje, visual y practico, se realiza en los juguetes. Y seguramente el juguete de propósito general más extendido es el Mecano. Un Mecano es un juguete que lleva al extremo el ingenio de un niño, y es uno de los juguetes que, incluso de mayores, despierta nuestro interés. Su principal atractivo es que PUEDES HACER LO QUE QUIERAS.

2. Objetivos

El objetivo principal de este proyecto es crear un mecano virtual con el que el usuario pueda manipular Piezas para realizar un Montaje, y que pueda manipular dicho Montaje según su deseo. Al mismo tiempo, se cubrirán los siguientes objetivos:

- Realizar el diseño de una aplicación cuyo interfaz con el usuario es principalmente gráfico.

- Profundizar en las herramientas OpenGL en la fase de desarrollo de la aplicación.

- Aglutinar aspectos técnicos y experiencias en un producto final y poder experimentar así lo que posteriormente realizaremos en el desarrollo de nuestra profesión.

- Respecto del proyecto concreto del que hablamos podemos decir que los objetivos son:

    1. Realizar un interfaz con el usuario lo más intuitivo y sencillo posible.

    2. Dotar de las herramientas de situación del punto de vista suficientes como para visualizar al detalle la zona deseada e incluso detectar problemas en el Montaje.

    3. Dotar de herramientas al conjunto del Montaje para poder visualizar y simular su funcionamiento como si estuviera en nuestra mano.

3. Especificación de requisitos

En este capitulo describiré la especificación de requisitos del proyecto Mecano Virtual, pretendiendo ofrecer una definición completa y global de la funcionalidad, a título de requisitos del usuario.

3.1. Introducción

A la hora de hacer una especificación de un producto, el responsable de realizarla se ha de poner en la piel del usuario final y ha tener un contacto muy próximo con él, para entender sus necesidades. Por ello, en la fase inicial de la realización de esta especificación, voy a suponer que el fabricante de un Mecano contrata mis servicios porque quiere suministrar, junto con el mecano físico, un software para poder simular que Montajes puede hacer con el modelo comprado y con los modelos superiores.

Así mismo, dado que este proyecto es más complejo que el que se puede realizar en el tiempo disponible para el proyecto final de carrera, la presente especificación acotará que contendrá esta primera versión del programa, y que no contendrá.

Esta especificación de requisitos esta dirigida al usuario del sistema.

3.2. Descripción general

A la hora de hacer esta descripción se han tenido en cuenta las principales acciones que se realizan con un Mecano, esto es :

- Crear un Montaje.
- Simular su funcionamiento.

Para crear un Montaje con un Mecano lo que principalmente se realiza es seleccionar Piezas de distintos tipos y montarlas según sus características y según la propia inspiración del constructor.

Para simular el funcionamiento de un Montaje lo que principalmente se realiza es observarlo de forma concienzuda reproduciendo las posibilidades propias de cada Pieza y observando su resultado en el conjunto del Montaje.

Todas estas operaciones se van a realizar de forma interactiva entre el usuario y lo que observa en la pantalla.

La descripción de estas operaciones sería como sigue:

= Crear un Montaje. Compuesta por:

- Seleccionar/Deseleccionar una Pieza.
    Se realizará clicando sobre la Pieza. En el momento que este seleccionada se remarcará la Pieza, cambiándola de color. Si se quiere deseleccionar la Pieza se pulsará la tecla Q, volviendo esta a ser del color inicial.
- Mover y orientar Piezas.
    La Pieza seleccionada se preparará para pasar al Montaje seleccionando la orientación deseada en el Montaje. Esto se realiza con el teclado, con las teclas X, Y y Z, cada una de ellas girará la Pieza en el eje que especifica.
- Montar la Pieza seleccionada en el Montaje.
    Se selecciona una de las Piezas del Montaje clicando sobre ella. Esta Pieza queda remarcada cambiando de color. Una vez que tenemos las dos Piezas seleccionadas (la unitaria y la del Montaje) pulsando la tecla P la Pieza inicial pasa a formar parte del Montaje en la orientación deseada y unida en la Pieza seleccionada del Montaje.

= Guardar un Montaje.

- Al pulsar la tecla S se salvaran las posiciones y orientaciones de todas las Piezas en un fichero determinado.

= Cargar un Montaje

- Al pulsar la tecla L se cargan las posiciones y orientaciones de las Piezas que quedaron grabadas al pulsar S.

= Simular el funcionamiento de un Montaje. Compuesta por:

- Selección del estado de simulación.
    Para seleccionar el estado de simulación se ha de clicar una Pieza del Montaje sin tener seleccionada ninguna pieza unitaria. Dicha Pieza se remarcará cambiando de color. Para volver al estado de Montaje solo hay que desseleccionar pulsando la tecla Q. La Pieza volverá a su color inicial.
- Aplicar estímulos al Montaje.
    Una vez que estamos en el estado de simulación y tenemos una Pieza del Montaje seleccionada podemos aplicarle estímulos rotativos o de desplazamiento, estos pueden ser con el ratón o con el teclado.

= Cambiar de posición el observador para ver mejor un Montaje o una simulación.

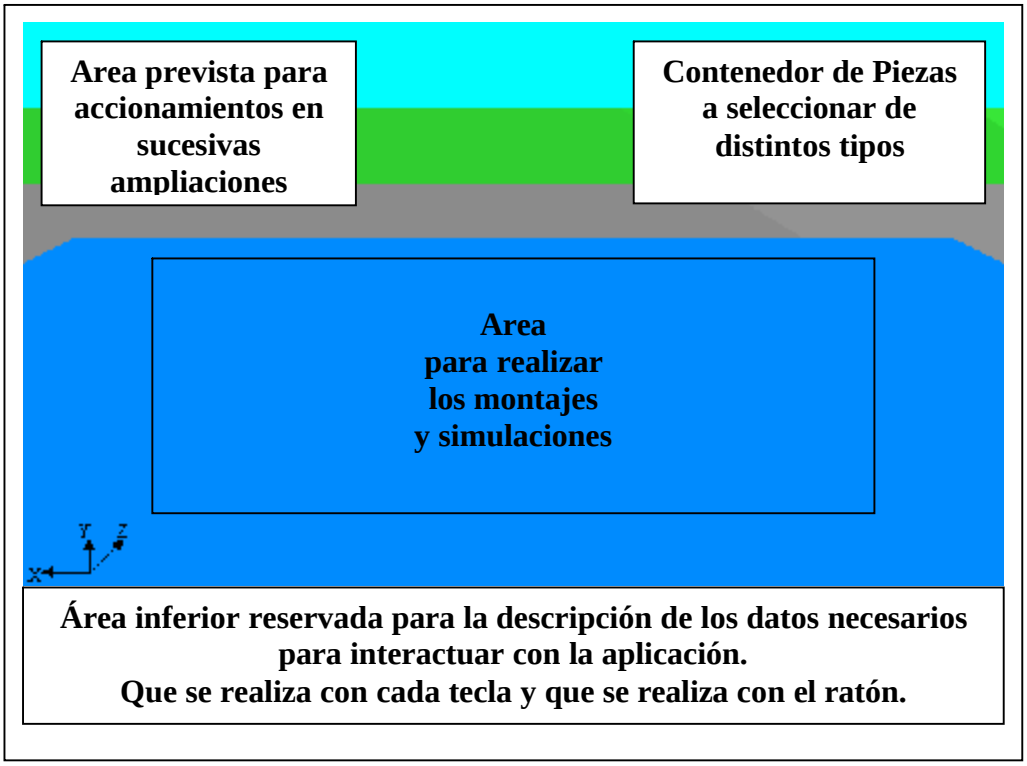
- Con las teclas de las flechas (Subir, Bajar, Derecha e Izquierda) y las teclas V y B (acercarse y alejarse) podemos variar la posición del observador respecto del Montaje.

Las principales operaciones que no realizará esta primera versión del programa serán:

- Crear una Pieza nueva.
- Incluir gran diversidad de Piezas distintas.
- Incluir diversos tipos de simulación, palancas, engranajes, etc.

3.3. Entorno de usuario

El entorno de trabajo simulará una mesa distribuida según la descripción siguiente :



3.4. Requisitos específicos

El presente proyecto se encuadra dentro de la asignatura de Informática Gráfica de la carrera de Engenieria Técnica Informática de Sistemes, por tanto se utilizan las herramientas propias de la practica de dicha asignatura, que son:

- GL4Java. Librería de tratamiento gráfico de la información.
- Java. Lenguaje de programación orientado a objetos. Concretamente la versión jdk1.3.1_04.

- Netscape. Visualizador de páginas html y applets. Concretamente la versión 4.78.
- Paquete Netscape.Security. Paquete que añade al Java las funciones necesarias para los permisos de leer y escribir en ficheros del disco duro.

3.5. Comentarios adicionales

Adicionalmente a los requisitos ya expuestos se prevén dos tipos de comentarios adicionales:

- Grabar / Leer ficheros en el Disco Duro desde un Applet

  Es obvio que para cumplir con los anteriores requisitos será necesario acceder al disco duro para guardar o leer información. Este acceso se realizará desde el Applet de la aplicación.

  Será el diseñador de la solución, o en su defecto el integrador, el que decidirá como se realizará esta operación. Tan solo deberá describir el proceso que ha seguido.

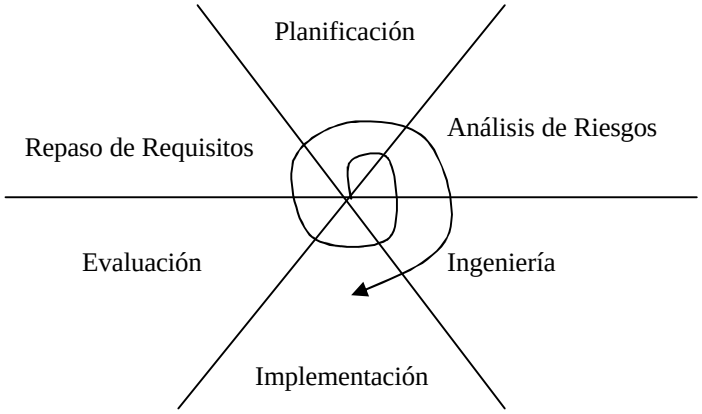
- Tipos de Piezas

  Igualmente pasa con el grupo de piezas con que parte el Mecano. Ya que no se dispone de una especificación y descripción de funcionamiento de la parte mecánica de este proyecto, más propia de una ingeniería mecánica, el diseño, o en su defecto la implementación, debe describir el tipo de Piezas inicial, el razonamiento seguido para ello y las ventajas detectadas.

4. Diseño e implementación

En este capitulo describiré las etapas de diseño, a partir de los requisitos del apartado anterior, e implementación de dicho diseño en el entorno descrito en el apartado 3.4..

Se ha seguido el ciclo de vida en espiral, refinando en cada interacción desde los requisitos hasta la ingeniería, tal y como muestra el siguiente esquema.



4.1. Introducción

Entramos en la etapa de diseño.

Nuestro objetivo es analizar toda la información que contienen los requisitos del proyecto y diseñar una solución para el contenido y trabajo de los datos que de ella se derivan, así como, no debemos olvidar que estaremos en un entorno gráfico, el entorno del usuario.

Uno de los pasos más importantes y extensos es el análisis de la información ya que de él se derivan todos los elementos en que se apoyará el resto del diseño.

4.2. Análisis de la información

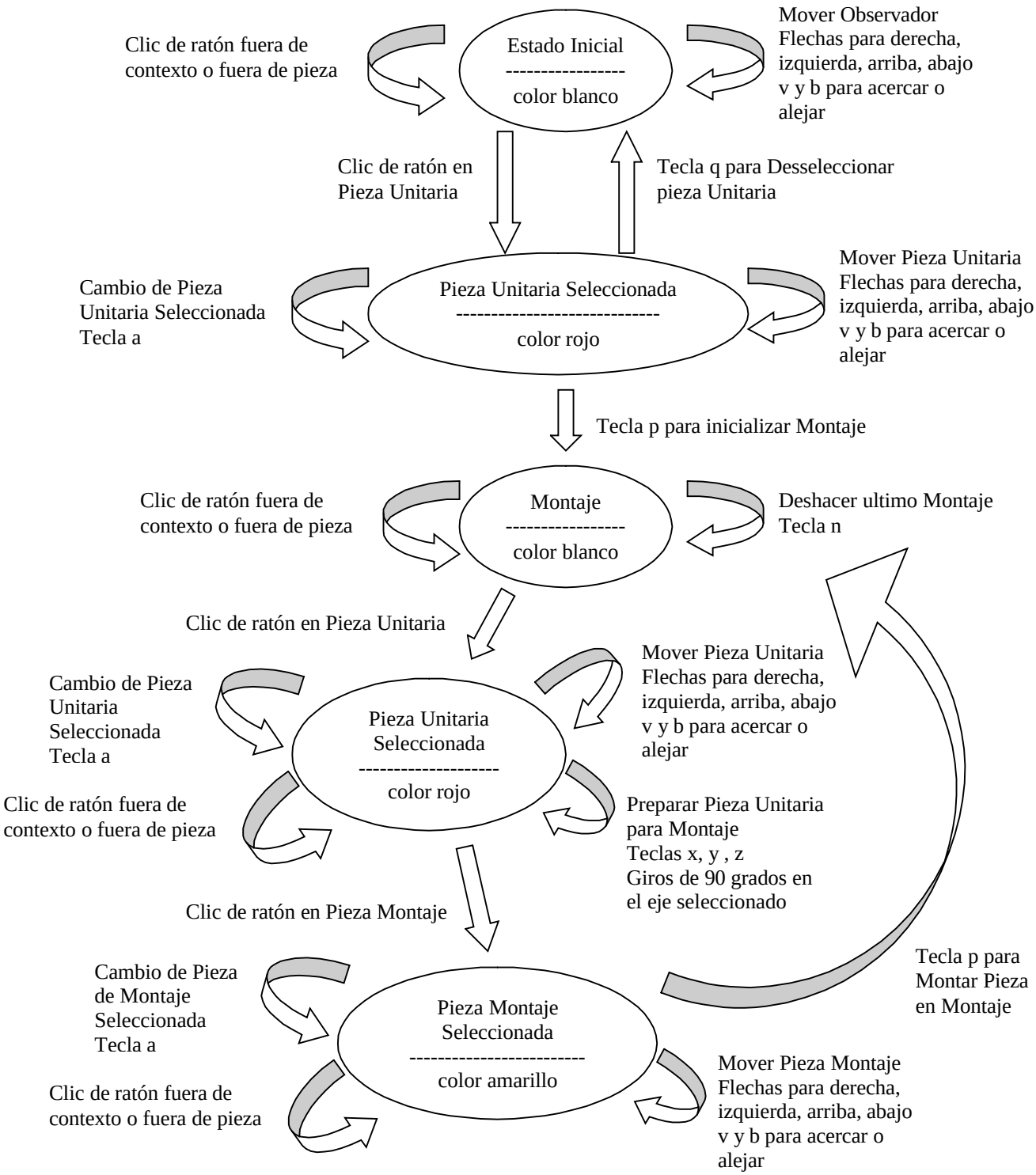
En este capitulo desgranaremos la información contenida en los requisitos del proyecto identificando las entidades externas e internas del sistema, así como las tareas principales a realizar. El resultado será la obtención de una serie de diagramas, especificaciones textuales y diagramas que describiremos en cada apartado.

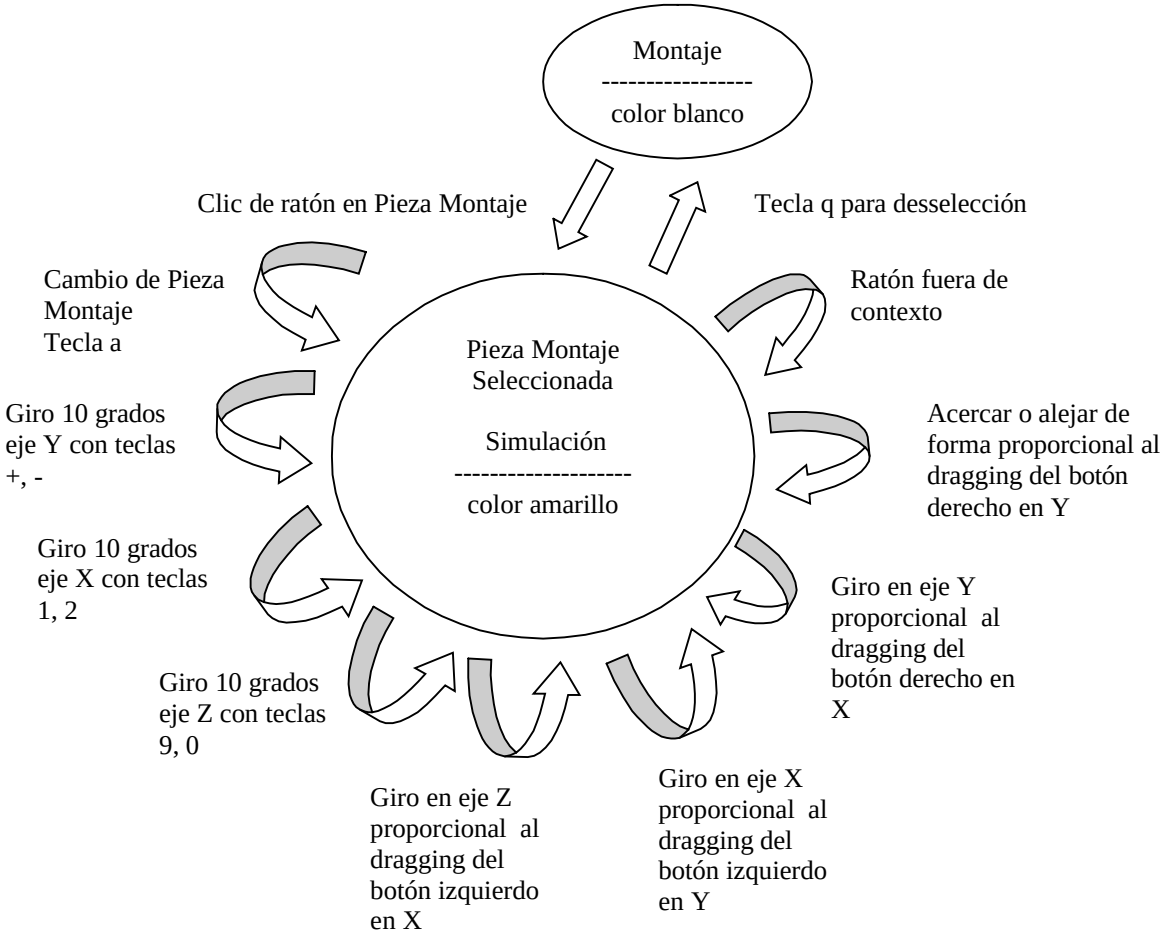
4.2.1. Diagrama de estados de la aplicación.

A partir de la información que contienen los requisitos se han desgranado los estados principales por los que pasa una sesión de trabajo típica. Estos estados se describen posteriormente detallando en su parte inferior como confirma el sistema que se consigue determinado estado.

En el primer esquema hay 5 estados. Los dos primeros son iniciales, montar primera Pieza o inicializar el Montaje, los tres siguientes realizan el trabajo de montar las sucesivas Piezas en el Montaje.

El estado tercero es de hecho el bisagra entre la operación de Montar o de Simular. Posteriormente se detalla el resto del diagrama de estados, incluyendo la operación de simulación.

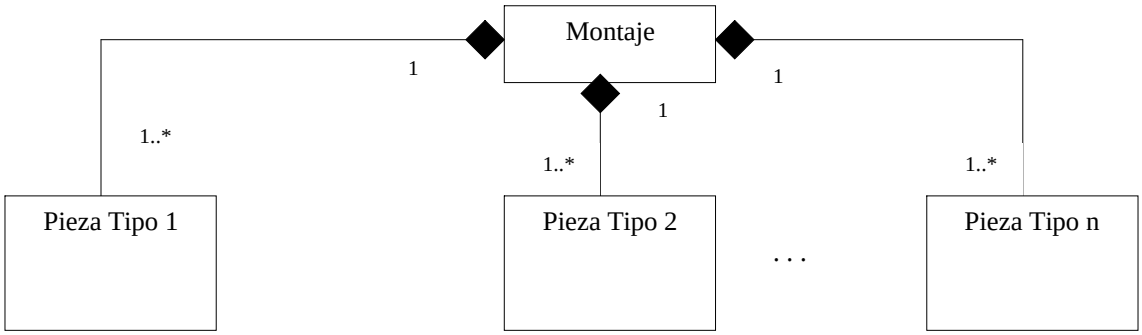

Clic de ratón fuera de contexto o fuera de pieza
Estado Inicial
color blanco
Mover Observador
Flechas para derecha, izquierda, arriba, abajo
v y b para acercar o alejar
Clic de ratón en Pieza Unitaria
Tecla q para Desseleccionar pieza Unitaria
Cambio de Pieza Unitaria Seleccionada
Tecla a
Pieza Unitaria Seleccionada
color rojo
Mover Pieza Unitaria
Flechas para derecha, izquierda, arriba, abajo
v y b para acercar o alejar
Tecla p para inicializar Montaje
Clic de ratón fuera de contexto o fuera de pieza
Montaje
color blanco
Deshacer ultimo Montaje
Tecla n
Clic de ratón en Pieza Unitaria
Cambio de Pieza Unitaria Seleccionada
Tecla a
Pieza Unitaria Seleccionada
color rojo
Mover Pieza Unitaria
Flechas para derecha, izquierda, arriba, abajo
v y b para acercar o alejar
Clic de ratón fuera de contexto o fuera de pieza
Preparar Pieza Unitaria para Montaje
Teclas x, y , z
Giros de 90 grados en el eje seleccionado
Clic de ratón en Pieza Montaje
Cambio de Pieza de Montaje Seleccionada
Tecla a
Pieza Montaje Seleccionada
color amarillo
Tecla p para Montar Pieza en Montaje
Clic de ratón fuera de contexto o fuera de pieza
Mover Pieza Montaje
Flechas para derecha, izquierda, arriba, abajo
v y b para acercar o alejar



4.2.2. Diagrama de clases del dominio

El diagrama de clases del dominio identifica, dentro del dominio del proyecto, cuales son las clases principales y que relación existe entre ellas. Mirando los requisitos tenemos:

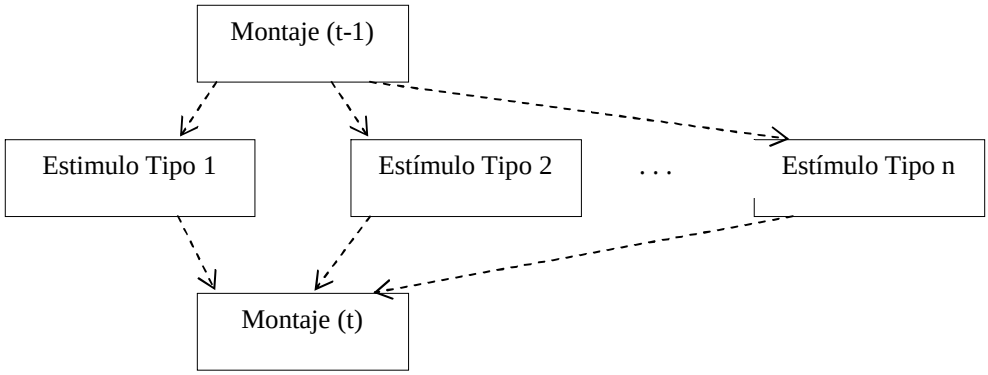
- Objetos del mundo exterior: Pieza (de distintos tipos)
- Objetos del dominio: Montaje



Se ha considerado un Montaje como una composición de distintas Piezas de diversos tipos. Igualmente, se ha considerado que las Piezas no tienen relación entre ellas si no es por la existencia del Montaje y que si el Montaje se destruye las Piezas en si

desaparecen, ya que no tienen sentido salvo para pertenecer a un Montaje, por lo menos dentro de este entorno.

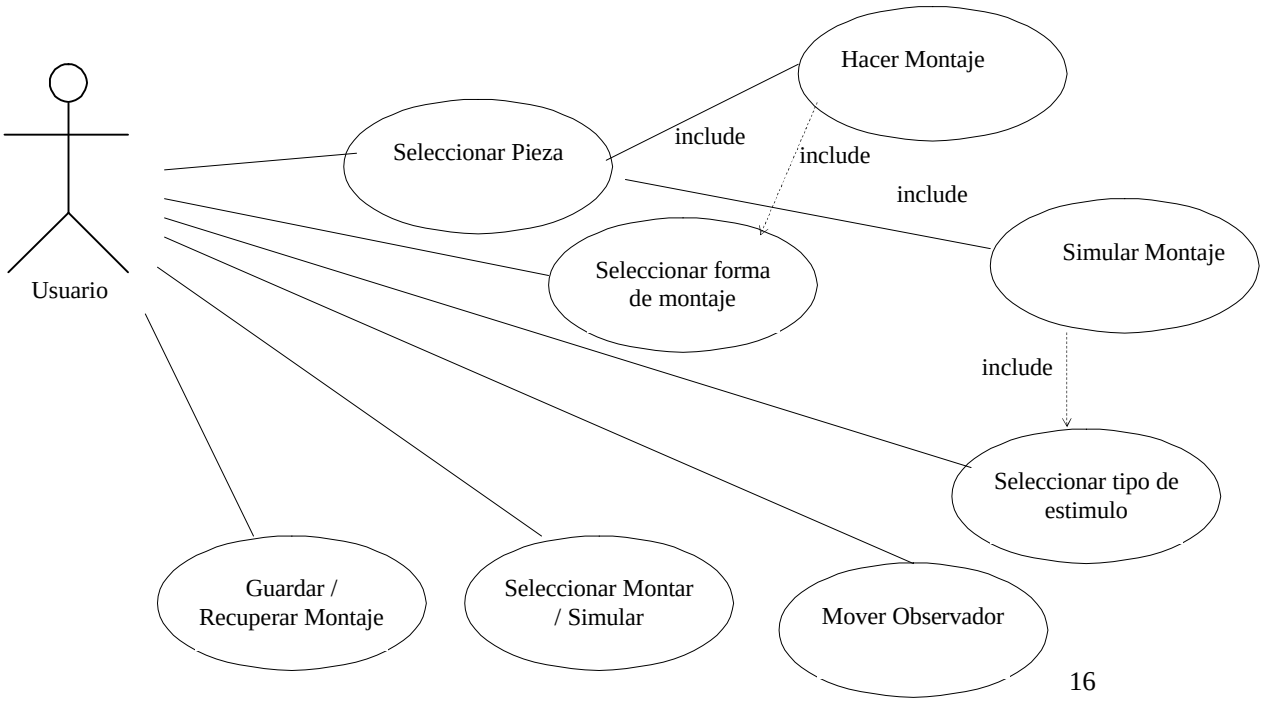
La simulación se describe como un proceso de modificación de la información inicial al aplicarse un estímulo. Posiblemente el estímulo si que sea un Objeto del Dominio. En esta primera vuelta del ciclo de vida solo se consideran dos tipos de estímulos (desplazamiento y giro) y es un poco pronto para ver si estos estímulos llegarían a representarse como objetos o no. Se deja para más adelante el ver si esto podría modificar el actual diseño.



4.2.3. Diagrama de casos de uso de los requisitos.

El diagrama de casos de uso de los requisitos pretende, una vez identificados los actores, protagonistas de la acción en el entorno, identificar que van a realizar dentro de dicho entorno, como lo van a usar, y ver la relación entre estos diferentes casos de uso.

El único actor identificado en los requisitos es el usuario.



Una vez identificados los casos de uso pasamos a ver sus relaciones.

- ❖ Los principales casos de uso son Seleccionar Pieza, Seleccionar Forma Montar , Seleccionar Tipo Estimulo , Seleccionar Montar / Simular y Guardar / Recuperar Montaje.

- ❖ Estos casos de uso inician otros por medio de relaciones include. Estos no son iniciados directamente por el actor principal aunque este interviene en todo momento en el proceso interactivo de dar datos.

4.2.4. Especificación textual de los casos de uso.

El paso siguiente, una vez identificados los casos de uso, es plasmar en una especificación textual más formal los requisitos del sistema. Dicha especificación se realiza para cada caso de uso identificado.

Caso de uso número 1: Seleccionar Pieza individual / Montaje

Resumen de la funcionalidad: a través del ratón se clica en la representación 2D de la escena en el lugar donde esta la Pieza que se quiere seleccionar. Si se clica en una Pieza del Montaje queda seleccionado este.
Papel dentro del trabajo del usuario: es un caso de uso básico.
Actor: **usuario**.
Casos de uso relacionados:
Precondición: puede haber una Pieza seleccionada o no.
Postcondición: hay Pieza marcada como seleccionada o Pieza de Montaje marcando como seleccionado.
Errores posibles: seleccionar una Pieza cuando ya hay una Pieza marcada como seleccionada, seleccionar una Pieza del Montaje cuando ya hay una Pieza del Montaje marcando como seleccionado.

Descripción detallada: El **usuario** selecciona mediante el ratón una Pieza suelta o una Pieza del Montaje. Si su intención es seleccionar una Pieza para montarla posteriormente, primero seleccionará la Pieza suelta y luego la Pieza del Montaje a la que se unirá.

Refinamientos posteriores:
El **usuario** puede desseleccionar una Pieza aislada o del Montaje mediante el teclado, tecla Q. La desselección está activa en cada uno de los turnos de selección.

El **usuario** puede cambiar de Pieza seleccionada o de Pieza del Montaje mediante el teclado, tecla A. El cambio esta activo en cada uno de los turnos de selección.

Caso de uso número 2: Seleccionar Forma Montaje

Resumen de la funcionalidad: por medio de las teclas X, Y y Z se gira la Pieza en cada eje.
Papel dentro del trabajo del usuario: es un caso de uso básico.
Actores: **usuario**.
Casos de uso relacionados:
Precondición: hay una Pieza suelta marcada como seleccionada
Postcondición: la Pieza suelta esta en la posición que se desea que se monte.
Errores posibles: la orientación seleccionada de la Pieza suelta sugiere conexión en una cara de la Pieza del Montaje ya ocupada.

Descripción detallada: El **usuario** selecciona mediante el teclado la orientación de la Pieza antes de montarla. Dicha selección se puede hacer con o sin tener seleccionada la Pieza del Montaje a la cual se unirá. Cada vez que se pulse una tecla girará la Pieza 90 grados en el eje seleccionado.

Refinamientos posteriores:

Caso de uso número 3: Hacer Montaje / Montar Pieza en Montaje

Resumen de la funcionalidad: por medio del teclado, tecla P, se encaja la Pieza seleccionada en el Montaje.
Papel dentro del trabajo del usuario: es un caso de uso básico.
Actor: **usuario.**
Casos de uso relacionados: Seleccionar Pieza y Seleccionar forma de Montaje.
Precondición: la Pieza suelta está seleccionada y en posición para ser montada y hay Pieza del Montaje seleccionada.
Postcondición : la Pieza forma parte del Montaje. No hay Pieza seleccionada.
Errores posibles: la orientación seleccionada de la Pieza suelta sugiere conexión en una cara de la Pieza del Montaje ya ocupada.

Descripción detallada: El **usuario** da la orden de que la Pieza seleccionada y orientada forme parte del Montaje.

Refinamientos posteriores:
Si la Pieza suelta seleccionada no está orientada pero la Pieza del Montaje seleccionada solo tiene una cara para ser

montada la Pieza suelta se montará en la cara libre de la Pieza del Montaje.

Caso de uso número 4: Mover Observador

Resumen de la funcionalidad: por medio de las teclas de las flechas, y de las teclas V y B se mueve la posición del observador, siempre mirando hacia el centro del Montaje.
Papel dentro del trabajo del usuario: es un caso de uso básico.
Actor: **usuario.**
Casos de uso relacionados:
Precondición: la posición del observador está en una posición inicial. La escena muestra como se ve desde la posición del observador. El Montaje está en el centro de la escena. No hay ninguna Pieza seleccionada ni suelta ni del Montaje.
Postcondición: la posición del observador ha cambiado según la orden dada. La escena muestra como se ve desde la posición del observador. El Montaje permanece en el centro de la escena.
Errores posibles: al dar la orden de bajar podríamos sobrepasar un límite que diera como resultado una imagen impropia de lo que se desea. Algo parecido puede pasar al acercarse al Montaje. Estudiar posibles errores en los otros sentidos.

Descripción detallada: El **usuario** selecciona mediante el teclado como quiere que cambie la posición del observador.

Refinamientos posteriores:

Caso de uso número 5: Guardar / Recuperar Montaje

Resumen de la funcionalidad: con las teclas S y L salvamos y recuperamos el Montaje.
Papel dentro del trabajo del usuario: es un caso de uso básico.
Actores: **usuario**.
Casos de uso relacionados:
Precondición: la sesión ha sido iniciada y ejecutada sin problemas.
Postcondición: las posiciones de las distintas Piezas de la escena quedan salvadas en un fichero, con las características del momento en que se pulsa la tecla S. Las posiciones de las Piezas son recuperadas del fichero con las características del momento de salvarlas.
Errores posibles: al leer el fichero no existe, esta incompleto o contiene errores de interpretación. Al crear el fichero hay problemas con la grabación del mismo. No se tienen permisos para poder acceder al los ficheros.

Descripción detallada: En el momento en que se pulsa la tecla S se salvan las posiciones de todas las Piezas del mecano, con las

características que se tienen en ese momento, si pertenecen o no al mecano, el punto de su centro de gravedad y su vector de orientación. En el momento en que se pulsa la tecla L se leen los datos del fichero y se actualizan las posiciones, orientaciones y características de las Piezas. Se continua en el estado de Montar.

Refinamientos posteriores:

Caso de uso número 6: Seleccionar Montar / Simular

Resumen de la funcionalidad: si en el momento de seleccionar una Pieza se selecciona una del Montaje, sin tener una seleccionada suelta, pasamos a la situación de simulación. Si desseleccionamos con la tecla Q pasamos a la situación de Montar.
Papel dentro del trabajo del usuario: es un caso de uso básico.
Actor: **usuario.**
Casos de uso relacionados: .
Precondición: inicialmente el programa tendrá seleccionado Montar. Según el transcurso de la sesión se puede tener seleccionado cualquier estado.
Postcondición: el estado habrá cambiado de Montar a Simular o de Simular a Montar.
Errores posibles:

Descripción detallada: Inicialmente la sesión empezará en el estado de Montar. Si se van seleccionando Piezas sueltas y Piezas del Montaje de forma alternativa el estado no cambia. Pero, si se pulsa una Pieza del Montaje sin haber seleccionado una Pieza suelta previamente, damos a entender que no queremos montar una Pieza y por tanto pasamos a estado de simulación. Para volver al estado de Montar solo hay que desseleccionar la Pieza con la tecla Q.

Refinamientos posteriores:
El Montar solo se puede realizar con Piezas de forma ortogonal, esto es, sin ángulos respecto de los ejes. Por esta razón al entrar en Simulación se guarda el estado de las Piezas en un fichero, y se recupera al desselecionar el estado.

Caso de uso número 7: Seleccionar tipo de estímulo / Simular Montaje

Resumen de la funcionalidad: Sobre la Pieza del Montaje seleccionada se aplican cuatro tipos de estímulos por medio del ratón o por medio del teclado. Esta Pieza transmite el efecto del estimulo inical al resto de las Piezas del Montaje.
Papel dentro del trabajo del usuario: es un caso de uso básico.

Actores: **usuario**.
Casos de uso relacionados: Seleccionar Montaje.
Precondición: estamos en el estado de Simular, esto es tenemos seleccionada una Pieza del Montaje.
Postcondición: se ve el efecto que causa aplicar el estímulo al Montaje a través de la Pieza seleccionada.
Errores posibles: pequeños desplazamientos del ratón pueden originar el efecto contrario al deseado. Al acercarse la Pieza al observador puede causar un efecto no deseado.

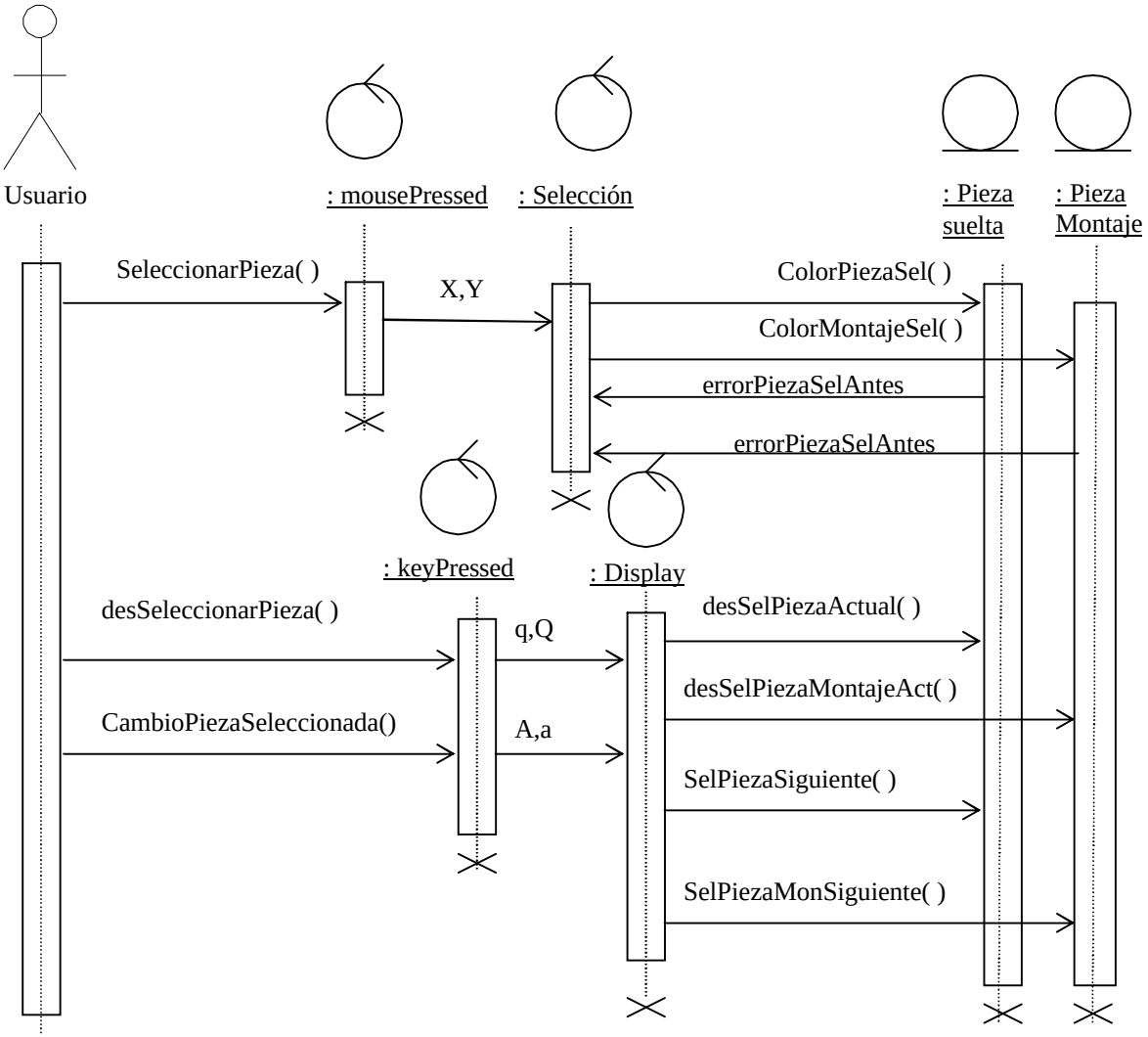
Descripción detallada: Todos los estímulos se pueden dar en los dos sentidos (a mas o a menos). Los cuatro tipos de estímulos son girar sobre los ejes X, Y o Z y acercarse o alejarse.
Si se aplican estos estímulos por medio del teclado son de un valor constante de giro o de distancia. Las teclas son +, - para el eje Y, 1, 2 para el eje X, 9, 0 para el eje Z y V, B para acercarse o alejarse.
Si se aplican estos estímulos por medio del ratón los desplazamientos o giros son más pequeños y se aplican de forma progresiva para dar más sensación de control del estímulo. Se utilizan las teclas derecha e izquierda del ratón y se controlan los desplazamientos en el eje X e Y de la escena para producir los mismos efectos que las teclas.
Con la tecla espacio se puede seleccionar otra Pieza del Montaje.

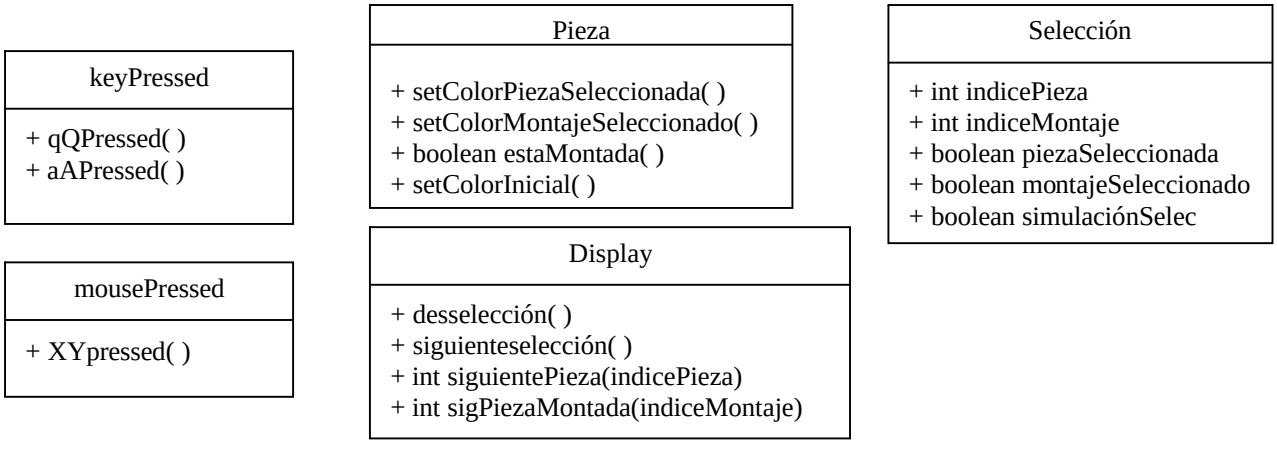
Refinamientos posteriores:
Con las teclas de las flechas podemos mover el Montaje hacia derecha o izquierda, arriba o abajo. Hay que tener cuidado de no exceder un mínimo para no obtener un efecto no deseado.

4.2.5. Diagramas de secuencia.

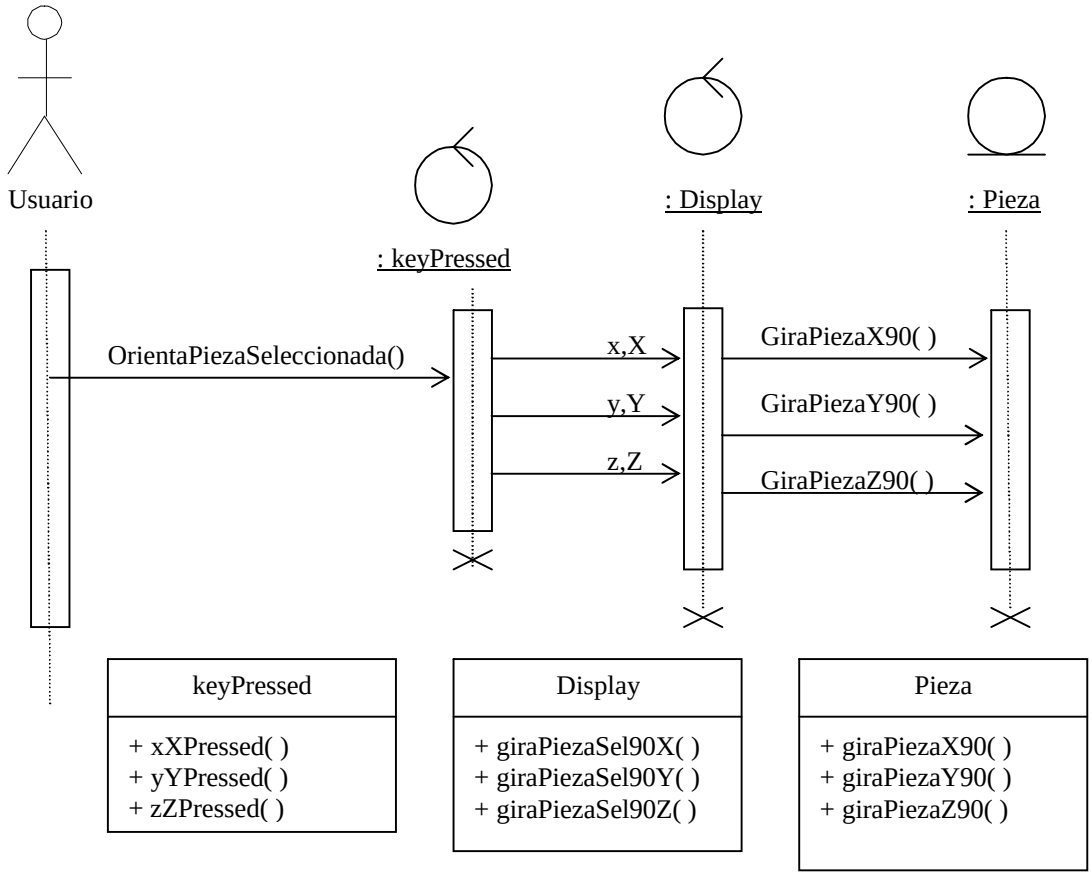
Los diagramas de secuencia pretenden describir las entidades que participan en cada caso de uso y ver como se interaccionan para conseguir su objetivo. Como hemos visto en el Diagrama de Estados y de Casos de Uso las principales secuencias son de Selección de Pieza, Selección de estímulo y Montar.

Caso de uso: **Seleccionar Pieza suelta/Montaje, Desseleccionar**

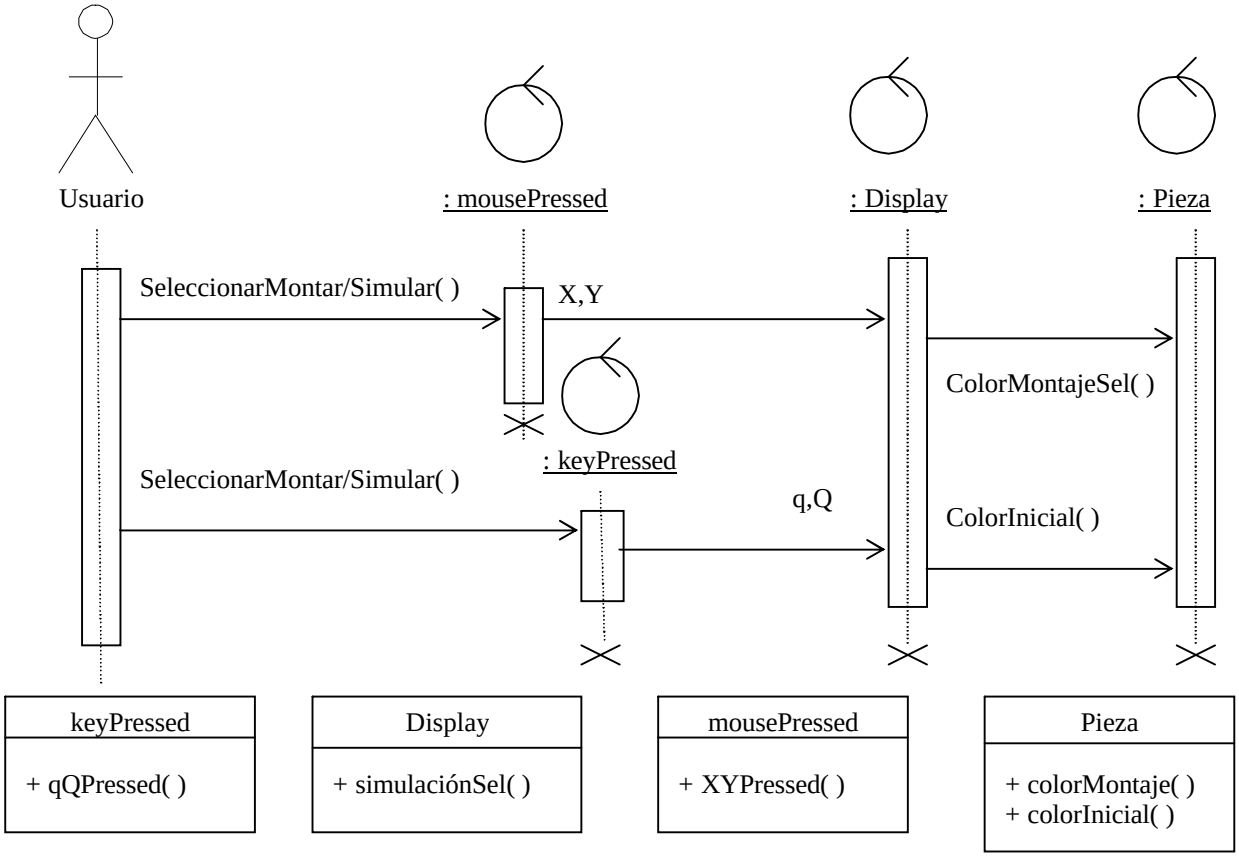




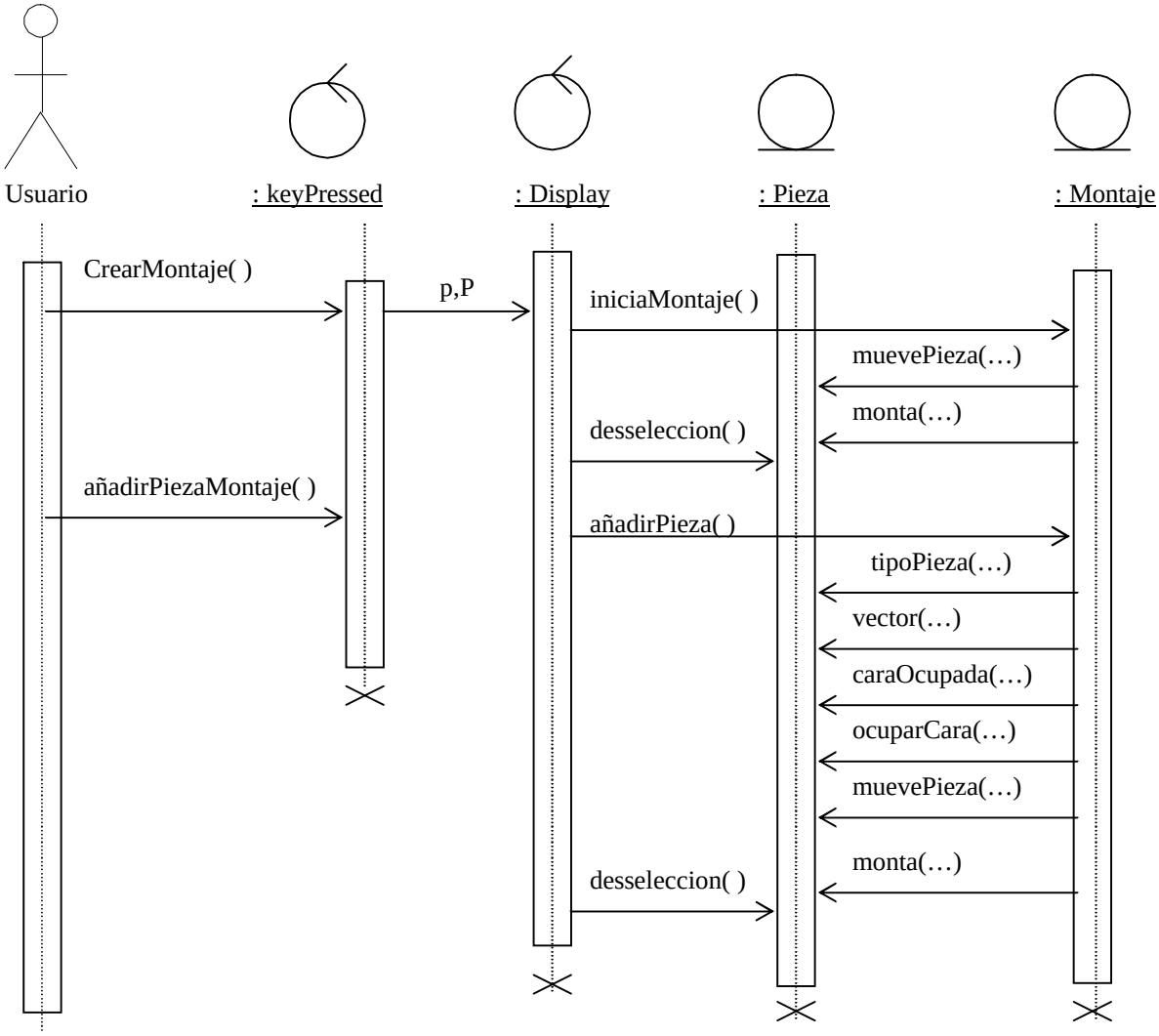
Caso de uso: **Seleccionar forma montar Pieza seleccionada**



Caso de uso: **Seleccionar Montar / Simular**



Caso de uso: **Hacer Montaje / Montar Pieza a Montaje**



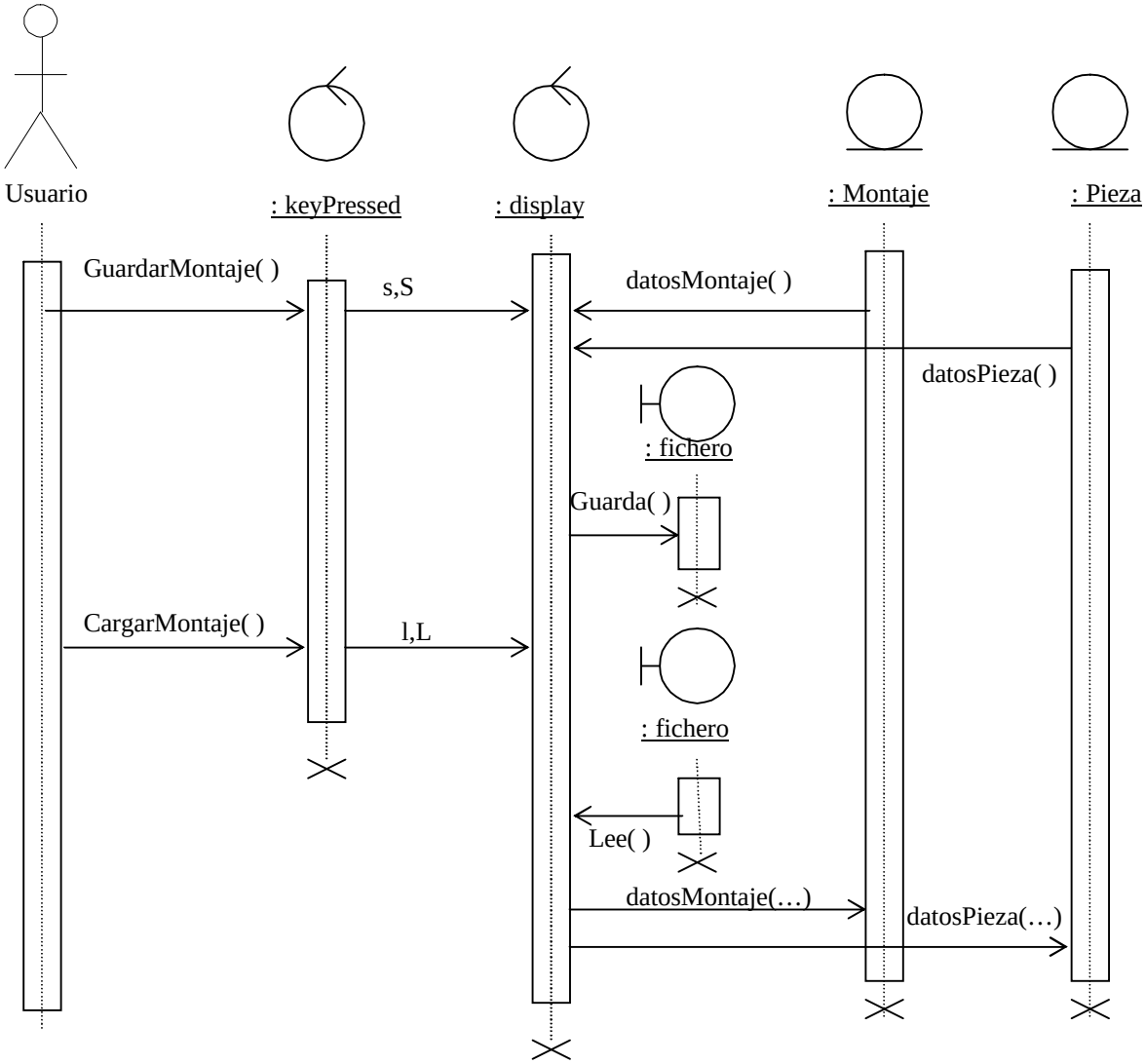
keyPressed
+ pPPressed( )

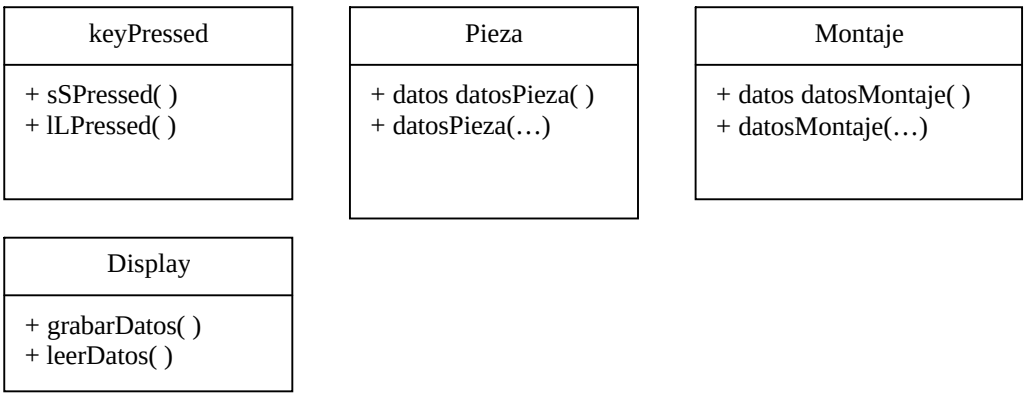
Display
+ desseleccionaPieza( ) + añadirPiezaAMontaje( )

Pieza
+ muevePieza(…) + monta( ) + tipo tipoPieza( ) + punto getCara(…) + punto getVector( ) + bool caraOcupada(…) + bool conexOcupada(…) + ocupaConexion (…) + ocupaCara(…)

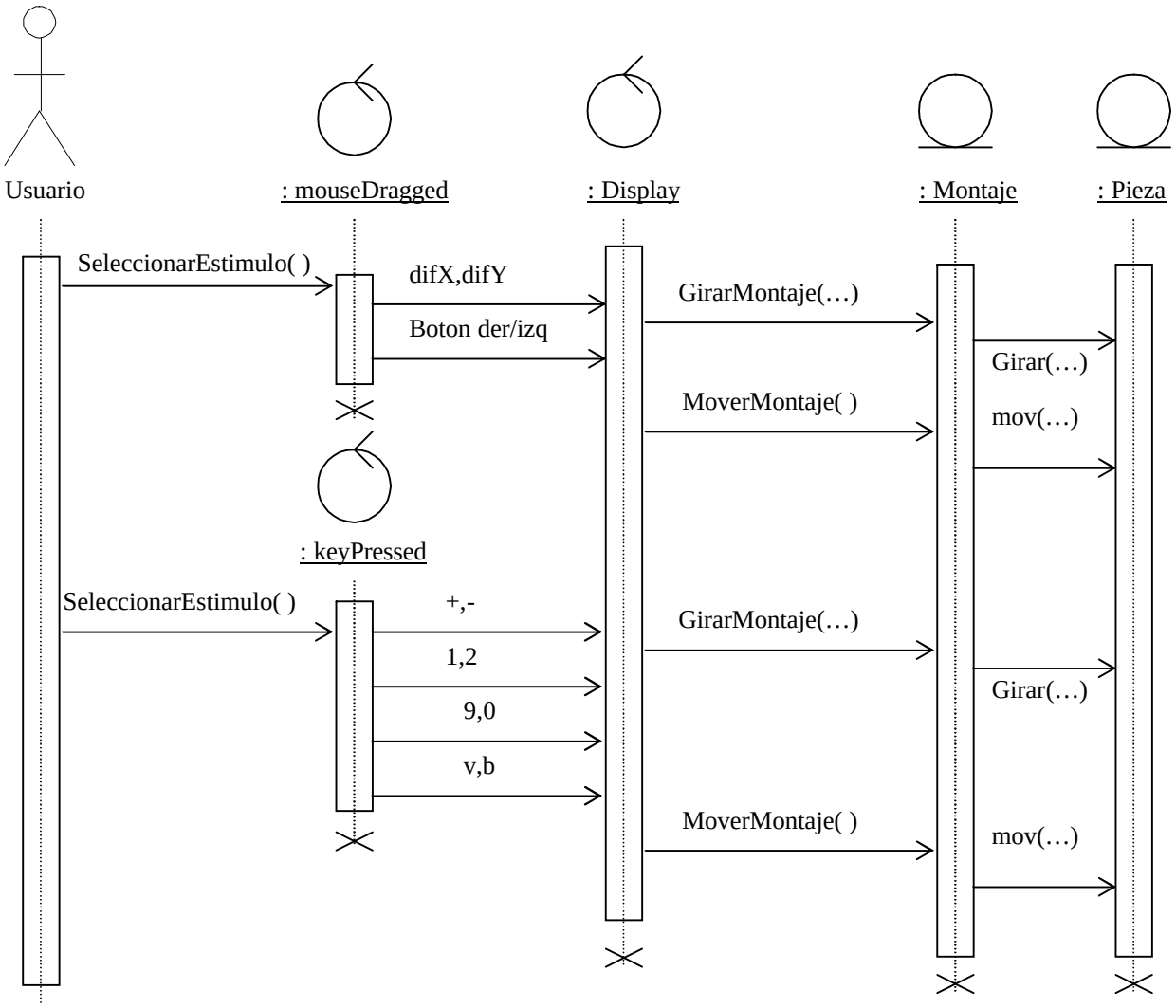
Montaje
+ iniciaMontaje(…) + AñadirPieza(…) + coincidenVectores?( ) + buscaCara( )

Caso de uso: **Guardar / Recuperar Montaje**





Caso de uso: **Simular / selección estímulo a Montaje**



mouseDragged
+ difXdifY( ) + botonDer( ) + botonIzq( )

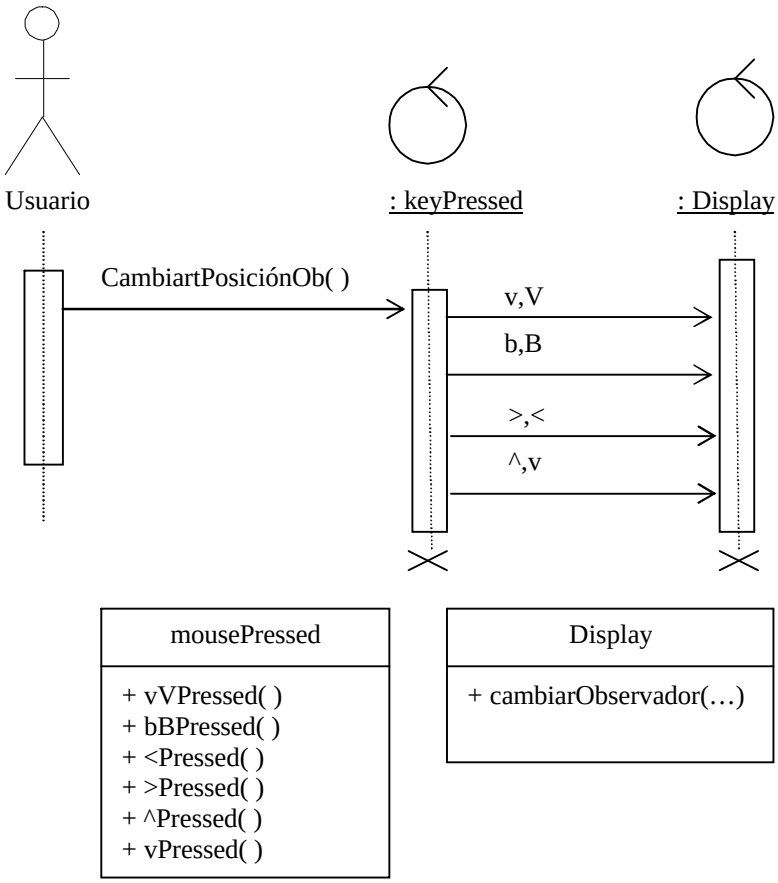
Display
+ detMouseBotonDer( ) + detMouseBotonIzq( ) + simularAngulo( ) + simularDist( )

Pieza
+ girarPieza(…) + moverPieza(…)

keyPressed
+ +Pressed( ) + -Pressed( ) + vPressed( ) + bPressed( ) + 1Pressed( ) + 2Pressed( ) + 9Pressed( ) + 0Pressed( )

Montaje
+ girarMontaje(…) + moverMontaje(…)

Caso de uso: **Mover Observador**



4.3. Diseño del Interfaz de Usuario

En el punto 3.3. de la Especificación de Requisitos se describe físicamente el Interfaz de Usuario. Dicha descripción, unida al apartado 3.2. Descripción General, deja bastante descrito como se va a interactuar con la aplicación.

Este concepto puede variar después del primer Prototipo, y después de la fase Evaluación, verificando la idoneidad de la solución y quedando patente si es lo intuitiva que se desea o si será necesario hacer algunos retoques.

El entorno donde se ejecutará la aplicación será NetScape ya que las librerías que se dispone para ejecutar el OpenGL así lo precisan.

Así pues el Entorno Gráfico contiene:

- un elemento de fondo lejano de color azul claro (simula el cielo)
- un elemento de fondo mediano de color verde (simula el campo)
- un elemento de fondo cercano de color gris (simula el suelo)

- un elemento de fondo de trabajo de color azul oscuro (simula tapete sobre mesa)
- ejes de ayuda de orientación en extremo inferior izquierdo (en negro)

- Piezas del mecano, de varios tipos, en posición inicial a la derecha

- descripción del funcionamiento en la parte inferior (que realiza cada tecla y el ratón en cada modo de funcionamiento)

4.4. Descripción de la Solución adoptada

La solución adoptada en el presente proyecto y su entorno gráfico esta basada en la experiencia de la practica realizada en la asignatura de IG. En el apartado 4.5. se describen los pasos seguidos para llegar a esta solución.

4.4.1. Encapsulamiento

Se parte del entorno OpenGL para Java diseñado en la asignatura, consistente en unas clases (geométricas) llamadas Punto3D, Vertex, Cara, Poligon y SolidFronteres y una clase contenedora de todas ellas ,Escena , a la cual se añaden las clases propias de los objetos físicos de la aplicación.

Todo ello dentro de la propia clase del Proyecto que enlaza las instrucciones propias del OpenGL para Java (GL4Java) con los servicios de escucha del teclado y del ratón y los servicios de dibujado.

La clase del proyecto extiende una clase especial de Java llamada Applet para poder ver todo el contenido de la escena en una página web. De esta forma el proyecto es visible con un visualizador de páginas web, típicamente el NetScape.

4.4.2. Contenedores de Información

Tenemos inicialmente la clase **Escena** que es un contenedor de todos los elementos a visualizar. Realmente, esta clase contiene un vector de un número máximo de SolidFronteres. Cada **SolidFronteres**, a su vez, es un contenedor de Caras y Vertexs, con el mismo tipo de implementación, vectores.

Las Caras ya son algo distintas, son vectores de Vertexs, pero tienen un color, y, como mínimo una normal para indicar por que lado son opacas. Finalmente un **Vertex** es un **Punt3D**, ósea, un punto de tres coordenadas, x, y, z , con tantas normales como a caras corresponda.

Debido a que una normal y un color son definidos por 3 coordenadas utilizaremos la clase Punt3D como tipo de dato para

ellos. Dentro de la clase SolidFronteres tenemos un servicio que realiza un SolidFronteres por escombrado de un Poligon una distancia determinada. Un **Poligon** lo podemos describir como un conjunto, en un vector, de Punt3D.

Normalmente cuando añadimos elementos a la Escena creamos nuevas clases que habitualmente tienen unos atributos típicos como ColorPieza, Profundidad, CentroDeGravedad, VectorOrientación y, por supuesto, el propio SolidFronteres del elemento si este es simple, o varios SolidFronteres si el elemento es compuesto.

Estas nuevas clases tienen unos servicios típicos como Constructor( ), para construir el elemento con parámetros por defecto, accesos de escritura de los atributos mencionados, ConstruyePieza( ), para construir el elemento con los atributos actualizados, y accesos de lectura de los atributos y del propio SolidFronteres del elemento, o de los diversos SolidFronteres si el elemento es compuesto.

Todo lo descrito anteriormente es genérico, falta concretarlo para el proyecto en curso. El Mecano Virtual, donde los elementos de la escena serán las Piezas del Mecano.

Como Pieza que es, la clase que lo describa tendrá servicios para poderla mover (según un vector y una distancia dados) y para poderla girar (según un ángulo dado).

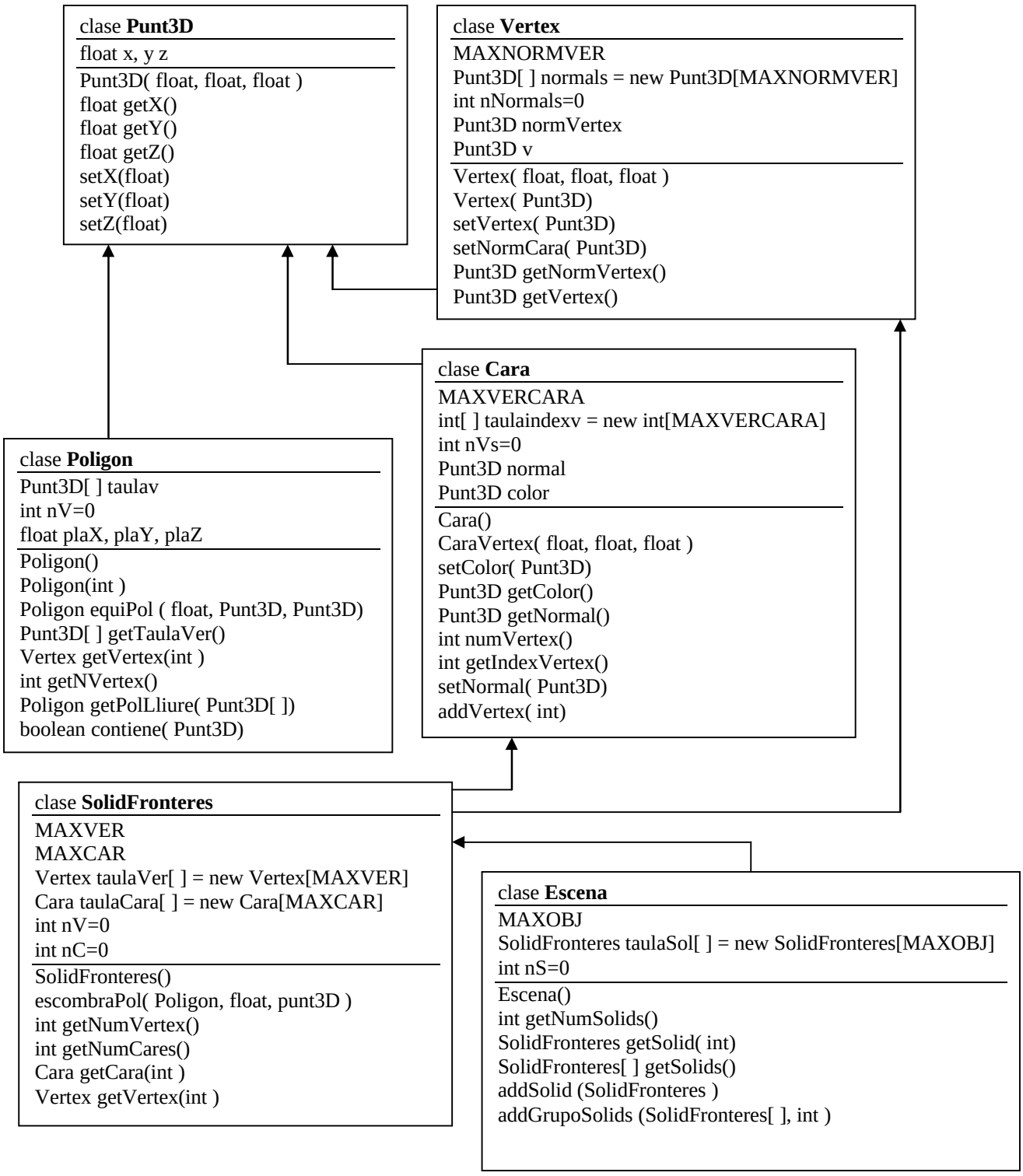
Según lo descrito en la fase de requisitos del sistema cada Pieza del Mecano puede o no pertenecer al Montaje que se está realizando, por tanto tendremos un atributo, en la propia clase de la Pieza, que será de tipo Booleano, que nos dirá si pertenece o no al Montaje (esMontaje). A su vez estarán los servicios de consulta y escritura de este atributo.

Igualmente cada Pieza tendrá por definición puntos donde se podrá conectar a otras Piezas y puntos donde se le podrán conectar a ella, estos puntos también son atributos de la propia clase, de tipo Punt3D, y también tendrá sus servicios de consulta y escritura.

Y por ultimo, ya que el Montaje se podrá guardar y recuperar, tendremos un accesor de lectura del centro de gravedad y el vector de la Pieza y un constructor de la Pieza a partir de estos mismos datos.

4.4.3. Estructura de clases contenedoras de la información gráfica a representar.

Seguidamente se va a representar la relación existente entre las distintas clases gráficas usadas en el proyecto. De cada clase se describen los atributos y los servicios de que dispone.



4.4.4. Estructura de la clases contenedoras de la información de cada Montaje y PiezaTipo del Mecano Virtual.

clase **PiezaTipo**

SolidFronteres PiezaTipo = new SolidFronteres()
int NM
int NH
int Tipo
Punto3D ColorPieza
Char TipoConex
float Profundidad
Punt3D CentroDeGravedad
Punt3D VectorConstruccion
boolean esEnMontaje
Punt3D[ ] puntoConexion
boolean[ ]conexOcupada
Punt3D[ ] puntosAConectar
boolean[ ] caraOcupada

PiezaTipo(int)

hazPiezaTipo() hazPiezaTipo(Punt3D)
setPuntoConstrucción(Punt3D)

setVectorConstrucción(Punt3D) Punt3D getVector()
setColorPieza(Punt3D) setColorPiezaInicial(Punt3D)
setColorPiezaSel(Punt3D) setColorMontajeSel(Punt3D)

monta() desmonta() boolean estaMontado()

ocupaCara(int) setOcupaCara(boolean[ ])
boolean estaCaraOcupada(int) Punt3D getCara(int)

ocupaConex(int) boolean estaConexOcupada(int)

char getTipoConex() setTipoConex(int)

setProfundidad(float)

setCDG (float, float, float) Punt3D getCDG()

Punt3D getPuntoConexion(int)
setPuntosConex(float[ ]) setCarasConex(float[ ])
setNorCaras(float[ ]) setVertices(float[ ])

SolidFronteres[ ] getPiezaTipo()

rotaX90() rotaY90() rotaZ90()

rotaPiezaX(double) rotaPiezaY(double)
rotaPiezaZ(double)

muevePieza(Punt3D, float)

String leeDatos() int obtenerDatos(int, byte[ ], int)

Teniendo en cuenta los casos de uso vistos en el diseño y las funciones vistas en ellos, paso a describir las clases de PiezaTipo y Montaje.

La clase PiezaTipo puede albergar varios tipos de Piezas. Inicialmente tendrá dos tipos, una Pieza prismática con 6 caras de conexión hembra (Tipo=1) y una Pieza en cruz con 2 caras de conexión macho (Tipo=2).

NM será el numero mayor de conexiones Macho, actualmente 2, y NH será el número mayor de conexiones Hembra, actualmente 6. TipoConex es un carácter para indicar que tipo de Pieza es, H para Hembra y M para Macho.

Hay unos vectores de Punt3D y de booleanos para saber los puntos de conexión de cada cara y si dicha cara esta ya ocupada.

También tenemos un booleano para saber si la Pieza esta montada en el Montaje o no.

El constructor por defecto es PiezaTipo(int) donde se inicializan los parámetros y se define de que tipo es.

Como podemos observar tenemos todo tipo de constructores a accesores a parámetros, entre otras cosas para poder guardar los datos en un archivo y poderlos actualizar desde un archivo.

Tenemos los servicios de rotar 90 grados y un ángulo determinado y el servicio de mover según un vector y un valor.

En el apartado 4.4.6. se describen las principales funciones. Allí podemos ver como funcionarían algunos de los servicios de esta clase.

clase **Montaje**

---

int NP
int NCM
int piezasMontadas
boolean esActivo
int [ ] [ ] MatrizConexiones
int [ ] piezasUsadas
int [ ] copPiezasUsadas

Montaje()
IniciaMontaje (int , Vector, Punt3D)
AñadePieza (int, int, Vector)

boolean coincidenVectores (Punt3D, Punt3D, Punt3D)

int buscaCara (int, Punt3D, Vector)

boolean existePiezaMontaje(int)

boolean buit()

simulaMontaje (char, float, Vector, int)

booraPieza(int)

boolean quedanPiezas()

boolean existePieza(int)

int siguientePiezaMontada(int)

setNPiezas(int)
setMatrizConexiones(int[ ] [ ])
setPiezasUsadas(int [ ])

String leeDatos() int obtenerDatos (byte[ ], int)

La clase Montaje alberga la información sobre el conjunto de Piezas que esta unidas haciendo un Montaje y las características de cómo se han unido estas Piezas.

NP es el número de Piezas máximo que puede haber en un Montaje, en el proyecto 10. NCM es el numero de caras máximo de una Pieza del Montaje, en la aplicación 6.

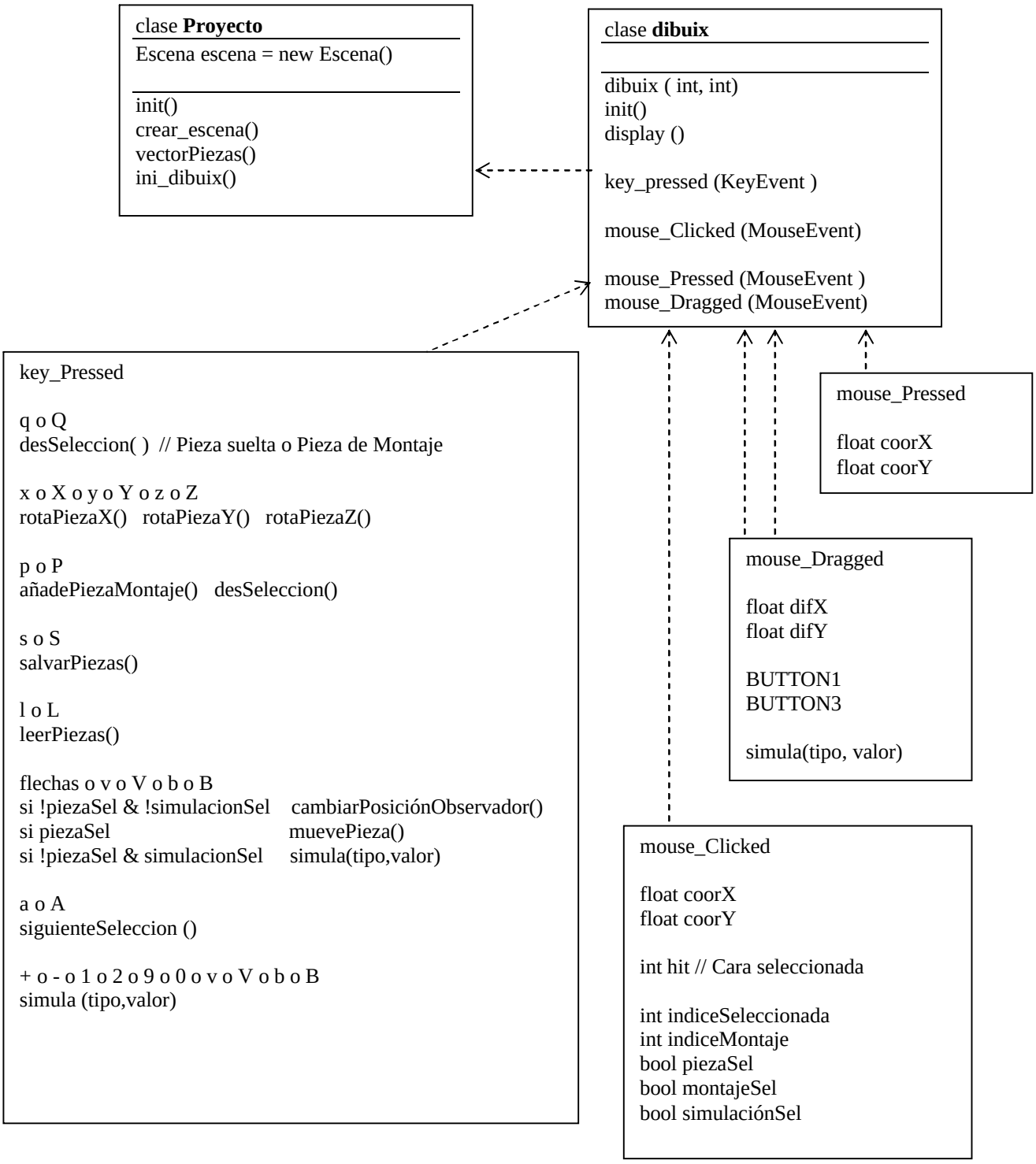
Con estos 2 valores creamos una matriz de NP filas y NCM columnas que inicialmente tiene valores por defecto de -1. En esta matriz anotamos las conexiones por medio de un número. Cada conexión tiene dos anotaciones. Si conectamos la Pieza 2 cara 3 con la pieza 5 cara 1, anotamos en la fila 2 columna 3 un 501 y en la fila 5 columna 1 un 203. Estas anotaciones dobles aumentan la eficiencia de la búsqueda de Piezas unidas cuando hacemos una simulación.

También mantenemos un vector con los números de las Piezas del Montaje, tal y como las vamos añadiendo. E igualmente un cantador de Piezas del Montaje.

Como podemos observar, al igual que en la clase PiezaTipo, tenemos todo tipo de constructores y accesores, entre otras razones, para poder guardar los datos en un fichero o leerlos de él. Igualmente podremos ver el funcionamiento de algunos de sus servicios en el apartado 4.4.6., descripción de las principales funciones.

4.4.5. Estructura del gestor de interficie del proyecto.

Paso a describir las clases y servicios que interactuan en la gestión de la interficie, esto es la comunicación con el exterior, del proyecto.



4.4.6. Descripción de la secuencia de funcionamiento de las principales funciones.

Seguidamente voy a describir el funcionamiento, de forma lo más formal y fácil de entender posible, de las principales funciones que se realizan.

- **Crear un Montaje**. Compuesta por:

  - Seleccionar/Deseleccionar una Pieza. Se realizará clicando sobre la Pieza. En el momento que este seleccionada cambiará de color para indicar que la operación se ha realizado correctamente. Si se quiere deseleccionar la Pieza se pulsará la tecla Q.

  1. Cuando ocurra un evento mousePressed se tendrán los valores X e Y del punto de la pantalla en que se ha clicado.

  2. Aprovechando la función de la biblioteca GL `gl.glRenderMode(GL_SELECT)` y haciendo un recorrido por todas las caras de la escena podremos saber en que Pieza se ha clicado.

  3. Si se ha pulsado en alguna Pieza y no hay Pieza seleccionada y la Pieza no esta montada se procede a cambiar el color de la Pieza seleccionada y a activar el flag de Pieza seleccionada. El numero de la Pieza seleccionada queda guardado en una variable global para disponer de el en otras funciones. Se llama indiceSeleccionada.

  4. Si se ha pulsado en alguna Pieza y no hay Pieza seleccionada pero la Pieza esta montada es que hemos pulsado en una Pieza del Montaje y nos disponemos a simular, por lo que cambiaremos el color de la Pieza seleccionada y activaremos los flags de Montaje seleccionado y simulación activa.

  5. Si se ha pulsado en una Pieza, ya había una Pieza seleccionada, no esta el Montaje activo y la Pieza esta montada es que estamos en proceso de montar una Pieza. Cambiaremos de color la Pieza del Montaje y activaremos el flag de Montaje seleccionado. El número de Pieza se guarda en una variable global para disponer de el en otros procesos. El nombre es indiceMontaje.

6. Cuando ocurra un evento keyPressed se detectará que se ha pulsado la tecla q o Q dejando el valor en una variable que tratará la función display.

7. En la función display se tratará el valor de la tecla pulsada y si es q o Q se realizará la función desSelección().

8. La función desSelección reestablece el color de la Pieza seleccionada al color inicial, desactiva los flags activos e borra los índices de las Piezas seleccionadas.

9. Cuando ocurra un evento keyPressed se detectará que se ha pulsado la tecla a o A dejando el valor en una variable que tratará la función display.

10. En la función display se tratará el valor de la tecla pulsada y si es a o A se realizará la función siguienteSelección().

11. La función siguienteSelección selecciona la siguiente Pieza no montada del vector de piezas si tenemos Pieza seleccionada activa y la siguiente Pieza del vector de Piezas montadas si tenemos Montaje seleccionado.

- **Orientar piezas.** La Pieza seleccionada se puede girar con el teclado para modificar la orientación de la misma. Esta será la orientación en que se montará en el Montaje. Las teclas serán X, Y y Z y cada una de ellas girará la Pieza en el eje que especifica.

1. Cuando ocurra un evento keyPressed se detectará que se han pulsado las teclas x o X o y o Y o z o Z dejando el valor en una variable que tratará la función display.

2. En la función display se tratará el valor de la tecla pulsada y si esta el flag de Piezas seleccionada activo se dará la orden oportuna a la Pieza que indique la variable global indiceSeleccionada.

- **Montar la Pieza seleccionada en el Montaje.** Después de haber seleccionado como se encaja la Pieza seleccionada y haber seleccionado en que Pieza del Montaje se va a encajar, en el momento en que se esta de acuerdo, se pulsará la tecla P para encajarla.

1. Cuando ocurra un evento keyPressed se detectará que se ha pulsado la tecla p o P dejando el valor en una variable que tratará la función display.

2. En la función display se tratará el valor de la tecla pulsada y si es p o P y el Montaje está vacío se iniciará el Montaje poniendo la Pieza seleccionada cerca del centro de la escena. Si la tecla es p o P y el flag de Montaje seleccionado esta activo se añadirá una Pieza al Montaje con los valores de las Piezas en las variables globales. Después de cada función desseleccionamos las Piezas tratadas.

3. Iniciar el Montaje consiste en mover la Pieza en un punto cercano al centro de la escena, poner el atributo de la Pieza como montada, poner el numero de la Pieza en el vector de piezasUsadas del Montaje e incrementar el contador de piezasMontadas.

4. Añadir una Pieza seleccionada a un Montaje es un proceso complejo. Empieza comparando los tipos de Piezas que se van a unir, ya que no está permitido que Piezas del mismo tipo se unan. Seguidamente se inicia un proceso de búsqueda de la cara de la Pieza del Montaje que tiene el mismo vector que el deseado para conectar la Piezas suelta seleccionada. Si se encuentra la cara se obtiene su punto de conexión y se mueve la Pieza seleccionada a este punto. Se informa a la Pieza seleccionada que está conectada y se rellena la matriz de conexiones con las uniones de las dos caras de las dos Piezas (dos anotaciones). Se dan por ocupadas ambas caras de ambas Piezas. Se anota la Pieza en el vector de piezasUsadas y se incrementa en contador de piezasMontadas.

❖ **Guardar un Montaje**.

- Al pulsar la tecla S se salvaran las posiciones y orientaciones de todas las Piezas en un fichero determinado.

1. Cuando ocurra un evento keyPressed se detectará que se ha pulsado la tecla s o S dejando el valor en una variable que tratará la función display.

2. En la función display se tratará el valor de la tecla pulsada y si es s o S se guardará la posición, el vector y las características principales y necesarias de todas la Piezas en un fichero de texto de nombre fijo.

❖ **Cargar un Montaje**

- Al pulsar la tecla L se cargan las posiciones y orientaciones de las Piezas que quedaron grabadas al pulsar S.

1. Cuando ocurra un evento keyPressed se detectará que se ha pulsado la tecla l o L dejando el valor en una variable que tratará la función display.

2. En la función display se tratará el valor de la tecla pulsada y si es l o L se modificarán la posición y las principales características de todas las Piezas según el contenido de un fichero de texto determinado.

❖ **Simular el funcionamiento de un Montaje**. Compuesta a priori por:

- La simulación es un estado distinto del de Montaje. Una vez que estamos en el estado de simulación podemos realizar desplazamientos y/o giros de las Piezas del Montaje a partir de una de las Piezas del mismo previamente seleccionada. Este proceso se puede hacer por medio del ratón o por medio de las teclas.

1. Cuando ocurra un evento keyPressed se detectará que se ha pulsado la tecla +, -, 1, 2, 9, 0, v, b, V, B o las flechas dejando el valor en una variable que tratará la función display.

2. En la función display se tratará el valor de la tecla pulsada y si es + simularemos un giro del Montaje respecto del eje Y un ángulo de 10 grados en sentido antihorario. Si es - simularemos un giro del Montaje respecto del eje Y un ángulo de 10 grados en sentido horario. Lo mismo pasa con las teclas 1 y 2 sobre el eje X , y las teclas 9 y 0 sobre el eje Z. Las teclas v y b moverán el Montaje 2 mm en el eje Z y las teclas de las flechas de izquierda a derecha y de arriba a abajo. Después de cada orden dada se realiza la simulación oportuna.

3. Una simulación lleva consigo varios pasos. Primero se realiza la simulación deseada en la Pieza del Montaje seleccionada. Se borra esta Pieza de una copia del vector de piezasUsadas y se recorre la matriz de conexiones empezando por la fila de la Pieza seleccionada. Mientras queden Piezas por encontrar de la copia del vector de piezasUsadas se recorrerán todas las columnas (caras de las Piezas) buscando conexiones. Cuando se detecte una conexión se conocerán el número de cada Pieza y la cara por la que están unidas. En la nueva Pieza se aplicará el movimiento, giro o ambos que se deduzca de la diferencia de cotas de los puntos de conexión de las caras conectadas. Al mismo tiempo se guarda el numero de la Pieza encontrada para posteriormente ir a su fila en

busca de nuevas conexiones, esto se realiza en un vector ya que una Pieza puede tener varias conexiones. Por ultimo se borra dicha Pieza de la copia del vector de piezasUsadas y realizamos otra iteración si aun quedan Piezas del montase por recorrer, o sea resituar.

4. Cuando ocurra un evento mouseDragged se tendrán los valores difX e difY del punto de la pantalla en que se ha clicado (mousePressed) y el actual.

5. A partir de esta diferencia de valores y la coincidencia de si se está pulsando la tecla del ratón izquierda o derecha se distribuyen las funciones de giro y desplazamiento con unos valores menores de ángulo y de distancia para que la sensación del control sea mayor. Solo se tiene en cuenta la mayor de las componentes del movimiento del ratón, si el valor absoluto de difX la mayor y tenemos pulsado el botón de la izquierda giramos el Montaje en Z en un sentido o en otro según sea positivo o negativo difX. Si es difY el mayor, con el botón izquierdo pulsado giramos en X. Con el botón derecho giraremos en Y al ser mayor difX y nos acercaremos o alejaremos al ser mayor difY. En cada iteración giraremos 5 grados y desplazaremos 0.75mm.

6. Cuando ocurra un evento keyPressed se detectará que se ha pulsado las teclas q o Q dejando un valor en una variable que tratará la función display.

7. En la función display se tratará el valor de la tecla pulsada y si es q o Q se aplicará la orden de desselección( ), comentada anteriormente, y que en este caso significa que salimos del estado de simulación.

❖ **Cambiar de posición el observador** para ver mejor un Montaje o una simulación.

- Con las teclas de las flechas (Subir, Bajar, Derecha e Izquierda) y las teclas V y B (acercarse y alejarse) podemos variar la posición del observador respecto del Montaje.

1. Cuando ocurra un evento keyPressed se detectará que se han pulsado cualquiera de las teclas de las flechas o v o V o b o B dejando el valor respectivo en una variable que tratará la función display.

2. En la función display se tratará el valor de la tecla pulsada y se variará la posición del observador según sea necesario atacando directamente la función

`glu.gluLookAt` de OpenGL, donde se le da a conocer las coordenadas del observador.

3. De hecho, la posición del observador se calcula cara vez según una distancia entre el observador y el centro de la escena y unos ángulos respecto de los planos XZ e YZ. Esta distancia y estos ángulos son los que variamos con las teclas de control del observador.

4.5. Descripción de los pasos seguidos hasta la solución actual

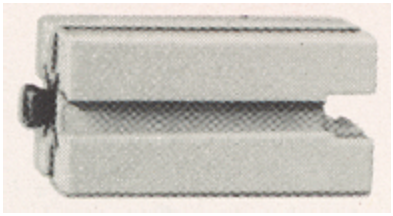
- **Respecto de las Piezas del Mecano.**

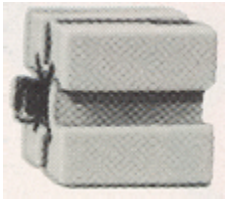
La solución que se presenta en esta memoria del proyecto ha ido evolucionando desde la concepción propia de la idea del proyecto hasta unos instantes antes de congelar el proyecto con la intención de presentar el estado actual en esta memoria.

El hecho de que el contenido propio de proyecto, esto es un Mecano, sea más propio de una carrera Mecánica de una carrera Informática ya ha creado alguno de los problemas.

Inicialmente se intentó trasladar al entorno del proyecto la idea simplificada de un Mecano del mercado, concretamente de la marca Fisher, el modelo FisherTechnic. Parecía que si se reproducía parte de este Mecano se podría llegar a Montar y Simular algunos de sus atractivos Montajes.

Sin mucho análisis, pero confiado en que este camino podría llevar hacia alguna meta cercana a la idea original, se empezó a trabajar en su Pieza base, esto es un prisma de base cuadrada, similar a un perfil de aluminio de los que se utilizan para el Montaje flexible de máquinas, líneas de Montaje, etc. Este prisma tiene 5 caras hembra, donde se podría acoplar hasta 9 Piezas como ella misma y 1 cara macho.





Inicialmente tome la simplificación de que por cada cara solo se podía conectar una Pieza, como si la pieza fuera un cubo.

De hecho el primer prototipo de solución contenía 5 Piezas como la primera y sirvió para ver la estrategia de Montaje y de simulación.

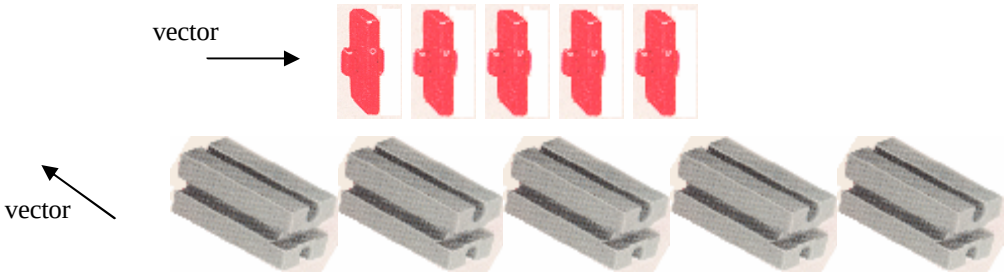
Sin embargo al intentar generalizar el sistema para mas Piezas se observaron problemas de gestión de las caras según fueran macho o hembra y sobre todo si la Pieza no daba ningún dato sobre su estructura. Llegado este punto se observo que la primera Pieza se podía considerar compuesta, o dicho de otra manera un submontaje. Como podemos intuir en el siguiente dibujo.

Así, pues, en el estado actual del proyecto, tenemos 5 Piezas de cada tipo simple.

La idea es que cualquier Pieza que se añada al Mecano ha de ser de tipo de conexión hembra, y siempre utilizaremos la Pieza doble macho para unir las Piezas. Si el proyecto avanzará se podrían generar submontajes y unir submontajes, pero para el tratamiento de la información siempre se estarían añadiendo Piezas doble macho a hembras.

Otra cosa que se ha aprendido es que una Pieza, como tal, tiene un vector y aun que se gire sobre un eje determinado y, en apariencia, quede igual la cara a la que se hace referencia es distinta.

Al entrar al programa del proyecto nos encontramos :



La forma de interpretar los vectores es de derecha a izquierda en las Piezas doble macho rojo, y de adelante a tras en las 6 hembra grises.

❖ **Respecto de la Clase PiezaTipo y otros tipos de Piezas.**

Inicialmente se pensaba que cada tipo de pieza que se generara sería una clase. De hecho, así era en el primer prototipo. Pero esto generaba mucha repetición de código ya que para realizar un mismo servicio en un objeto o en otro había que hacerlo de forma independiente. Esto llevó a generalizar la clase e incluir el atributo Tipo que define como es la pieza.

Intuyo que esta idea puede durar un tiempo y que al aumentar la complejidad de las piezas puede optarse por una jerarquía de clases según

las características de las piezas. Pero casi seguro que lo que puede provocar mas cambio es la inclusión de otros tipos de estímulos en la simulación. Aunque en este punto no he llegado a ver que esto pudiera ser una clase, mas bien una forma de clasificar las piezas.

- **Respecto de grabar/leer ficheros desde un Applet.**

Ya desde la especificación previa se consideró que podría ser oportuno el poder guardar las situación de las Piezas y del Montaje, o subMontajes, para poderlos recuperar mas adelante o en otra sesión. De hecho, se comentaba que el fichero sería único.

Durante la realización del primer prototipo llego el momento de probar esta posibilidad y se observo que no era posible por problemas de seguridad del sistema. Desde un Applet no es posible leer un fichero o guardar un fichero si no se tienen una serie de permisos adecuados, incluso desde el propio disco duro del usuario.

En este punto se chocó con una de las diferencias que hay entre los dos principales visores de páginas web o documentos html, el Internet Explorer y el NetScape.

Las primeras pruebas no dieron resultado. Aunque funcionaban con un pequeño programa, al intentarlas incluir dentro del prototipo el resultado no era aceptable. Hubo que recurrir a un proceso especifico para NetScape que paso a detallar.

Son necesarias dos cosas para completar este proceso: las clases NetScape Java Capabilities API y la herramienta de firmado SignTool 1.1. Las primeras son necesarias para alterar el código fuente del Applet e incluir ciertas llamadas a funciones para garantizar los privilegios requeridos, y la segunda para crear un certificado para firmar Applets y firmar y empaquetar los ficheros con las clases.

Paso previo. Crear contraseña para acceso a base de datos de NetScape.

> Dentro del NetScape Communicator se pulsa en la el botón de Seguridad de la barra de herramientas, se elige Passwords y se pulsa en el botón Establecer Passwords para crearlo.

Paso 1. Crear un certificado de prueba para firmar Applets con SignTool

> Una vez cerrado el Communicator se ejecuta SignTool con las opciones -G y -d. La primera es para dar un nombre al certificado y la segunda para indicar el directorio donde se encuentra la base de datos de certificados y las claves, osea donde estan los ficheros cert7.db y key3.db, típicamente en c:\Archivos de programa\Netscape\Users\Javier
>
> C:\Proyecto IG\signtool -G javier -d c:\archiv~1\netscape\users\javier

Durante la ejecución de la herramienta se piden una serie de datos como la organización, el departamento, etc, también se pide el password y se genera el par de claves publica y privada, el certificado y el propio certificado firmado, y añade el nombre dado a la base de datos. Como resultado tenemos los ficheros x509.raw y x509.cacert que utilizaremos mas adelante.

Paso 2. Crear un Applet con privilegios de lectura y escritura de ficheros en el disco duro.

En nuestro código que genera el Applet debemos importar las clase de la API de NetScape sobre seguridad.

```
import netscape.security.*;
```

Y para leer/crear/escribir ficheros tenemos que anteponer a las instrucciones propias de apartura del fichero la siguiente llamada.

```
PrivilegeManager.enablePrivilege("UniversalFileAccess
");
```

Es aconsejable, aunque molesto, revocar los privilegios concedidos después de la lectura o escritura del fichero en cuestión, con la siguiente llamada.

```
PrivilegeManager.revertPrivilege("UniversalPropertyRe
ad");
```

Durante la ejecución del Applet, el NetScape informa de la petición de estos privilegios y el riesgo que conlleva aceptarlos, si el origen del Applet no es fiable, y pide confirmación al propio usuario. Si se revoca en cada operación es un poco molesto, por tanto en el código del presente proyecto no se revocan.

Paso 3. Firmar y empaquetar los ficheros con las clases.

De nuevo SignTool nos facilitará la faena. Tenemos que ponerle todas las clases en un directorio y el programa se encargará de generar el fichero comprimido de todas las clases y de firmarlo. Precisará de las opciones -k, -d y -Z. La primera es para dar un nombre, -d ya lo hemos comentado antes y la ultima es para señalar el nombre que se le quiere dar al fichero comprimido y la clase origen del Applet. Evidentemente durante el transcurso del firmado pide el password de la dase de datos.

```
C:\Proyecto IG\signtool -k javier -d c:\archiv~1\netscape\users\javier -Z
Proyecto_IG.jar Proyecto_IG
```

Obtendremos un fichero Proyecto_IG.jar firmado.

Paso 4. Incrustar el fichero jar en la página web.

El directorio donde tengamos la página web debemos tener el fichero jar firmado, no se deben tener los ficheros de las clases.

El documento html debe tener el acceso al Applet de la siguiente manera.

```
<applet CODE="Proyecto_IG.class"
archive="Proyecto_IG.jar" WIDTH=700
HEIGHT=500></applet>
```

Para que otro usuario pueda acceder a los privilegios de los ficheros ha de tener el fichero x509.cacert e instalarlo o un fichero resultado de la exportación del certificado en el NetScape.

- **Uso de la potencia del Guardar y Leer Ficheros.**

Una vez resuelto el problema de guardar y leer ficheros desde un Applet se pudieron añadir ciertas funciones al sistema que le dieron cierta potencia y flexibilidad. Estas son:

Volver a situación inicial. Si en algún momento se quiere volver a la situación inicial si necesidad de parar la aplicación se puede realizar cargando un fichero que tiene almacenada esta situación.

Restaurar situación de montaje después de simulación. En la situación actual del proyecto solo se pueden añadir Piezas a un Montaje si las Piezas de este tienen sus vectores paralelos a los ejes X, Y o Z. Una forma fácil de poder continuar el Montaje después de la simulación es salvar la situación en que se inicia la simulación y recuperarla posteriormente.

Interprete para repetir proceso. Si hacemos un sistema que simule la secuencia de teclas pulsadas en un determinado orden, almacenadas en un fichero, podemos utilizar este sistema para varias cosas, como por ejemplo, para formar al usuario de cómo se usa el software, o para enseñar como se monta un determinado Montaje, etc

Función Deshacer. Se puede mantener un fichero, o varios, con los n estados últimos y así dar opción a volver a tras en la secuencia de ordenes dadas.

- **Respecto de la selección de piezas.**

La selección de piezas fue unos de los hitos difíciles de conseguir, y que cuando ya casi se tenia conseguido, se vio la forma de realizarlo utilizando las funciones OpenGL.

En realidad es deshacer todo el proceso que realiza el OpenGL para poner en una pantalla 2D los distintos vértices de una escena 3D. Se trata

de saber si hay una Pieza en la recta que ira desde la coordenada del observador, y que pasara por un punto de representación 2D igual a los valores X e Y clicados por el ratón. En caso de que hubiera varias Piezas se desearía saber la más cercana al observador.

El proceso que se realiza en las funciones int DoSelect(int, int) y render(int) es el siguiente:

> DoSelect devuelve un entero que representa la entidad más cercana al observador con una pirámide de visión que tiene el mismo vértice que la aplicación (el observador) y tiene una base donde el punto X Y clicado por el ratón esta en el centro y unos lados muy pequeños.
>
> ```
> glu.gluPickMatrix(x, (viewport[3]-y), 5.0f, 5.0f,
> viewport);
> ```
>
> Ponemos la opción GL_SELECT en la función `gl.glRenderMode` para recorrer los objetos que se deseen de la escena pero que esto no influya en la imagen a visualizar.
>
> ```
> gl.glRenderMode(GL_SELECT);
> ```
>
> Y se recorren los elementos de la escena cuantificando los que se desee detectar. En nuestro caso contamos las Piezas ya que cuando seleccionamos no distinguimos entre las caras, pero se podría contar las caras y posteriormente tratamos para saber que pieza es.
>
> ```
> gl.glLoadName(objectCount);
> objectCount++; // Cada Pieza o cada Cara
> ```
>
> Las primeras piezas tienen seis caras y las segundas piezas 14 caras.

4.6. Instalación del sistema

Para que el proyecto funcione hay que seguir los siguiente pasos para instalar el software adecuado:

- Instalar el NetScape.
- Añadir las librerías de GL4Java tanto al NetScape como al compilador de Java que se utilice.
- Crear un directorio con los ficheros Proyecto_IG.html y Proyecto_IG.jar

Para realizar estas operaciones se dispone, conjuntamente con este documento, de una copia de estos ficheros distribuidos por directorios.

Si queremos que la aplicación se pueda compilar debemos:

- Instalar JDK 1.3.1_04 o posterior
- Copiar la librería java40.jar en el mismo directorio de las librerías de GL4Java
- Modificar el autoexec.bat con la línea:

  set CLASSPATH=.; c:\jdk1.3.1_04\lib\Gl4Java.jar; c:\jdk1.3.1_04\lib\swing.jar; c:\jdk1.3.1_04\lib\java40.jar

4.7. Pruebas del sistema

Para certificar la robustez del sistema frente a errores de funcionamiento o de mala manipulación se han provocado los siguientes errores obteniendo la respuesta que se describe para cada error:

- Mala manipulación del Usuario

Pruebas del Sistema Errores de manipulación del Usuario Errores usando el ratón	
Prueba	Resultado
Clicar fuera del área	Correcto
Clicar en los bardes del área	Correcto
Clicar en los bordes de las piezas	Correcto
Hacer doble click en la Pieza	Correcto
Hacer Dragging de fuera del área a dentro	Correcto
Hacer Dragging de dentro del área a fuera	Correcto
Hacer un Dragging muy corto	Correcto
Hacer un Dragging muy largo	Correcto

Pruebas del Sistema Errores de manipulación del Usuario Errores usando el teclado	
Prueba	Resultado
Probar con Mayusculas	Correcto
Ctrl mas tecla	Correcto
Alt mas tecla	Correcto
Varias teclas a la vez	Correcto
Numeros	Correcto
Escape	Correcto
Return	Correcto
Retroceder	Correcto

- ✓ Comentarios a los resultados.
  Se ha obtenido la respuesta esperada. Dragging de fuera a dentro no hace nada. Clicar cerca de la pieza detecta pieza. Doble click como si fuera click. Dragging de dentro a fuera sigue haciéndose la operación. Las teclas 5 y 6 tienen funciones no documentadas, consideradas especiales, 6 hace representación por fronteras y 5 por sólidos. Si se tiene una tecla pulsada y se pulsa otra hace lo que corresponda a la tecla, si una tecla con función se mantiene pulsada hace la operación repetidamente.

- Error en funcionalidad

Pruebas del Sistema Errores de Funcionalidad Errores en Montaje	
Prueba	Resultado
Montar Macho Macho	Correcto
Montar Hembra Hembra	Correcto

Pruebas del Sistema Errores de Funcionalidad Errores en Simulación	
Prueba	Resultado
Deshacer	Correcto
Simulación	Correcto

Prueba secuencia aleatoria	Resultado	Comentario	Prueba secuencia aleatoria	Resultado	Comentario
7p87p67p	Correcto	Según lo previsto	7zp0y7p17p8y1p2yz8p	Correcto	Según lo previsto
2p32p12p	Correcto	Según lo previsto	7yp07p50p9xy0p6xy0p80p9z0p	Correcto	Según lo previsto
6xp16p4x6p74p	Correcto	Según lo previsto	5p1y5p3z5p	Correcto	Según lo previsto
1yp71p9z1p6xx1p	Correcto	Según lo previsto	0yp5zy0p6y0p36p1z5p	Correcto	Según lo previsto
2p72p8y2p6xz2p47p94p	Correcto	Según lo previsto	1yzzp71p8zzx1p	Correcto	Según lo previsto
3yp7z3p27p	Correcto	Según lo previsto	5zp0y5p3zzy5p	Correcto	Según lo previsto
7xxxp37p8yyzp4y8p17p61p	Correcto	Según lo previsto	8yp2xyz8p4x8p9xzy4p6zzy2p	Correcto	Según lo previsto
2zyzp82p18p91p3z9p5yxy3p63p	Correcto	Según lo previsto	1xp8yzyxyz1p71p5zx1p	Correcto	Según lo previsto

- ✓ Comentarios a los resultados.
  Si hubiera un error en el fichero de simulación se cargaría el fichero de Inicio.

- ❖ Información no adecuada en ficheros

Pruebas del Sistema Errores de manipulación de Ficheros Errores en Lectura	
Prueba	Resultado
No existe fichero	Correcto
Contenido de fichero no lógico	Correcto
Fichero incompleto	Correcto
Contenido corrompido	Correcto
Contenido de mas	Correcto
Datos desordenados	Correcto

Pruebas del Sistema Errores de manipulación de Ficheros Errores en Escritura	
Prueba	Resultado
Falta espacio	Correcto
Mismo nombre	Correcto
Conversión de datos	Correcto

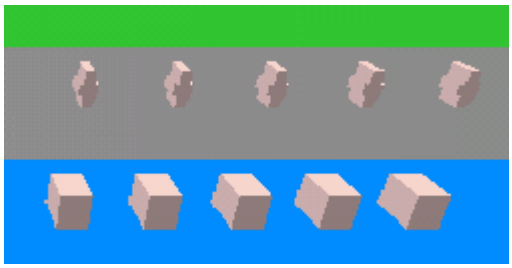
- ✓ Comentarios a los resultados.
  Si no esta el fichero de simulación o el fichero de el programa sigue funcionando. Si hay un tipo de dato que no coincide o el fichero está incompleto el programa sigue funcionando pero es mejor volver a empezar, se aconseja hacer inicio (tecla i). Si el fichero tuviera información de mas, pero la necesaria fuera correcta no pasaría nada. Datos correctos pero desordenados no paran el programa pero no se aconseja continuar, mejor inicio. Los errores dejan comentario en la consola de Java. Se considera que los ficheros generados en la misma sesión son correctos, esto limita la posibilidad de fallo a la opción L de leer fichero. Esta opción se ha ligado con Inicio en caso de error de lectura. El fichero de Inicio esta protegido contra escritura o modificación.

4.8. Manual de Usuario (Ejemplos prácticos)

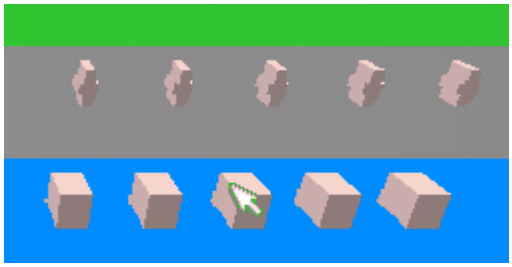
Siguiendo los siguientes pasos se puede empezar a coger una cierta soltura con el funcionamiento del Mecano. En el anexo 1 hay un ejemplo de uso completo.

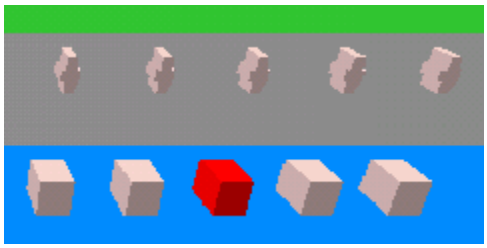
- ❖ Seleccionar una Pieza unitaria. Seleccionar orientación para tener en el Montaje. Cambiar de Pieza. Iniciar Montaje.

1. Situación Inicial. Piezas del lado derecho de la pantalla.

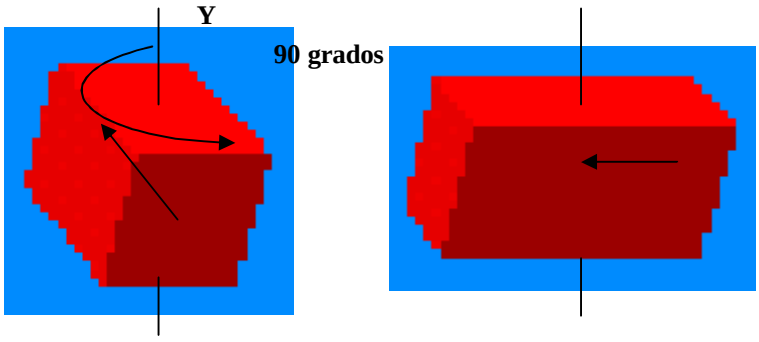


2. Con el ratón situarse en la pieza deseada y clicar el botón izquierdo. La pieza se coloreará de color rojo para indicar que es una pieza unitaria (no del montaje) seleccionada.

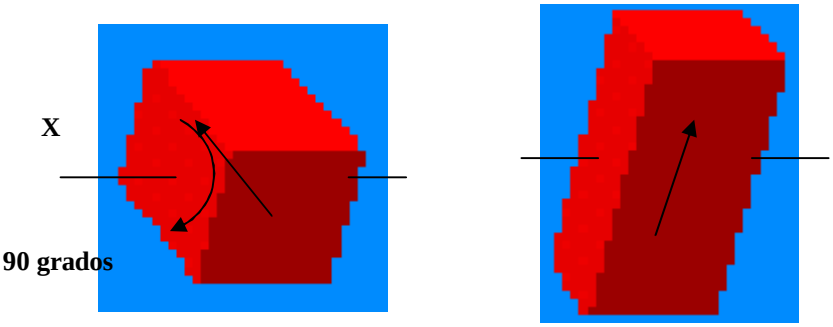




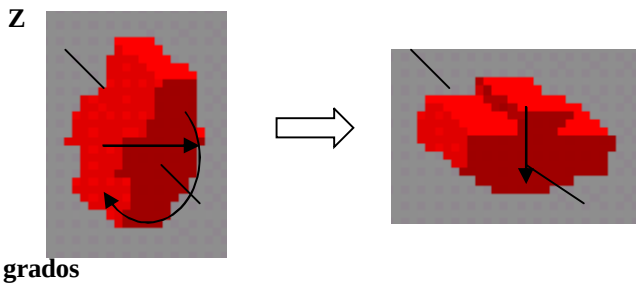
3. Con el teclado escogemos la situación que se desea para la pieza en el montaje. Si pulsamos Y tenemos:



Si pulsamos X tenemos:



Si pulsamos Z tenemos:

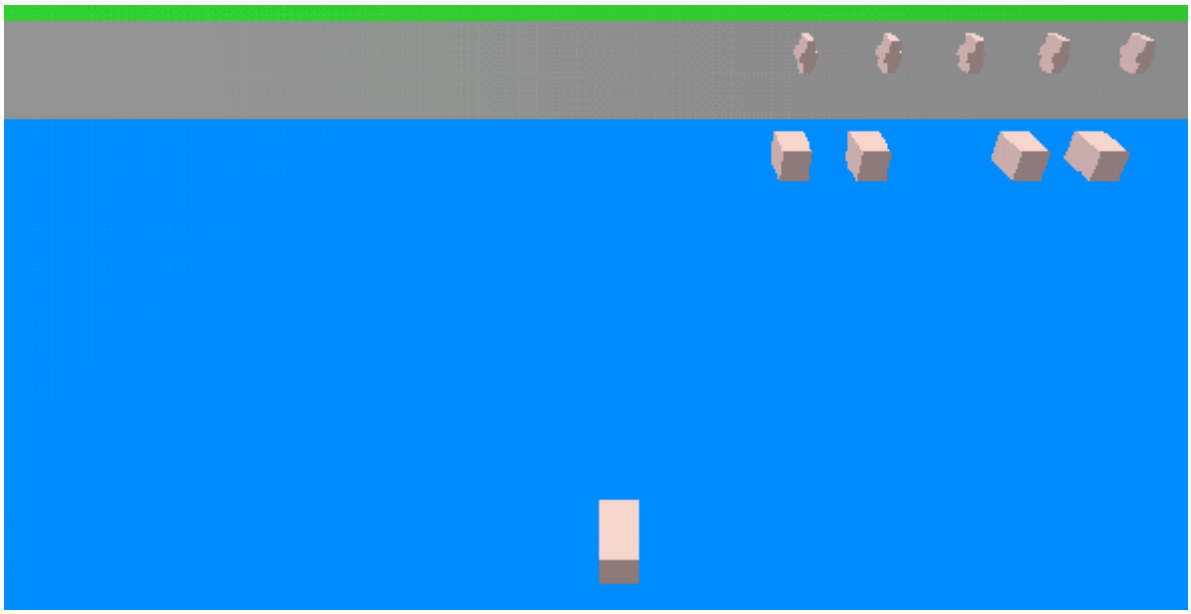


90 grados

Si pulsamos Q la pieza seleccionada cambia de color al inicial y queda desseleccionada. La pieza queda en la posición que estaba.

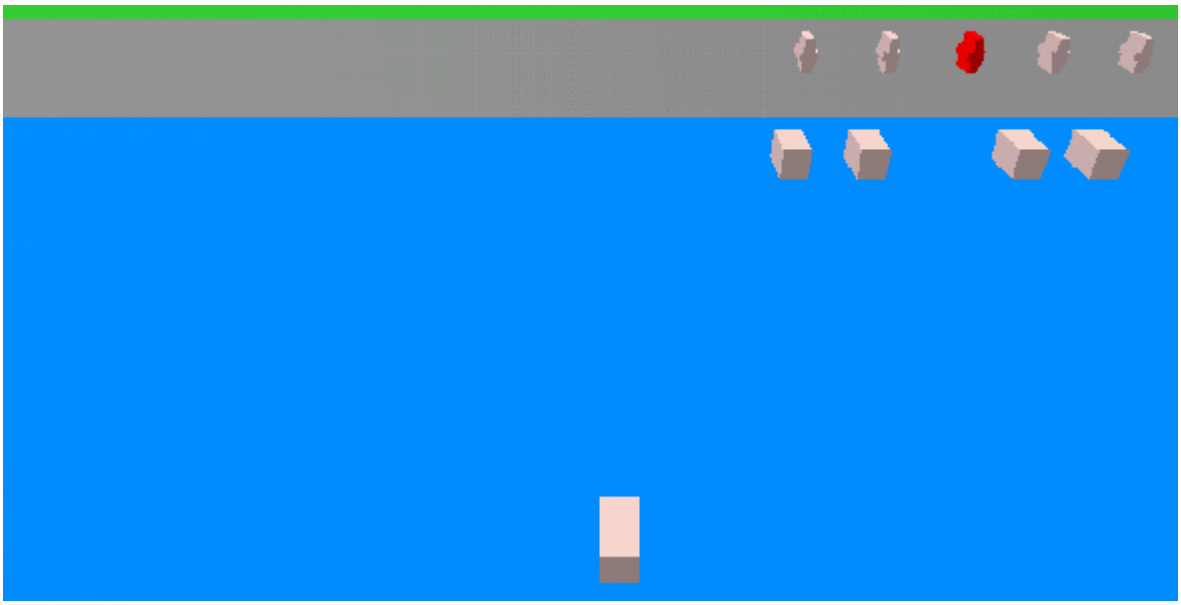
Si pulsamos A la pieza seleccionada queda desseleccionada seleccionándose la siguiente en el orden de creación que no este en el montaje.

Si pulsamos P la pieza seleccionada pasa al Montaje inicializándolo. El NetScape nos pide permiso para almacenar en el disco duro el estado actual, por si quedemos volver atrás. La pieza toma su color inicial en el Montaje (centro de la escena)

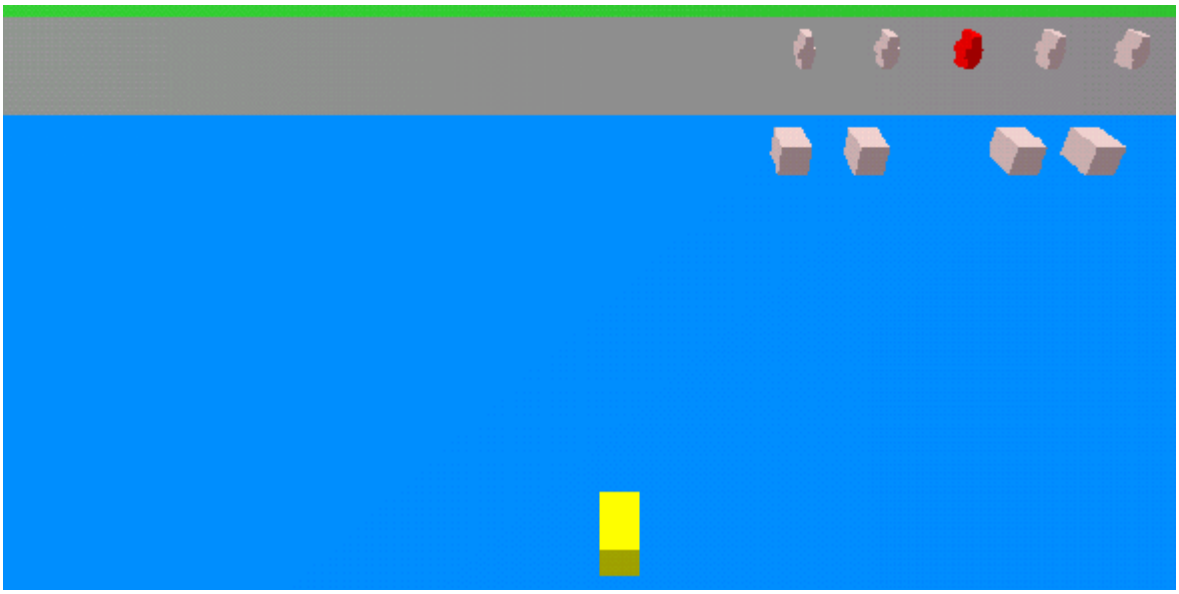


- Seleccionar una Pieza unitaria. Escoger posición para montar. Seleccionar una pieza del Montaje. Añadir pieza a Montaje. Deshacer ultimo montaje.

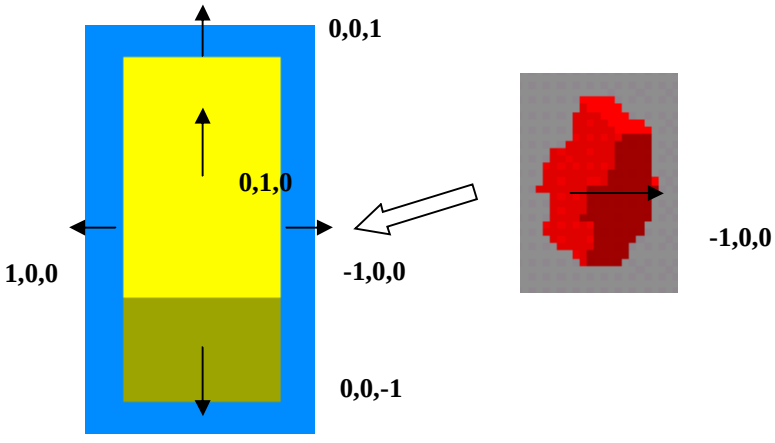
1. Las dos primeras acciones, seleccionar una pieza y escoger posición de montaje, son iguales a la fase anterior, con una salvedad importante, el tipo de pieza ha de ser distinto. Luego tenemos una pieza doble macho seleccionada y coloreada de rojo.



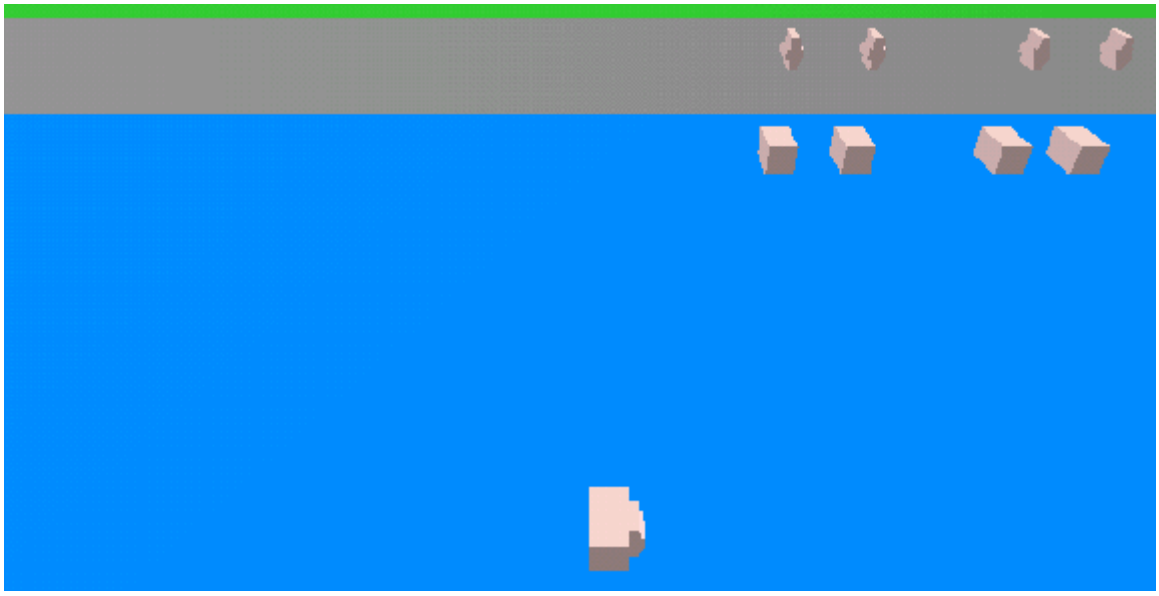
2. Con el ratón seleccionamos la pieza del Montaje a la que conectaremos la que hemos seleccionado. La pieza del Montaje seleccionada se coloreará de amarillo.



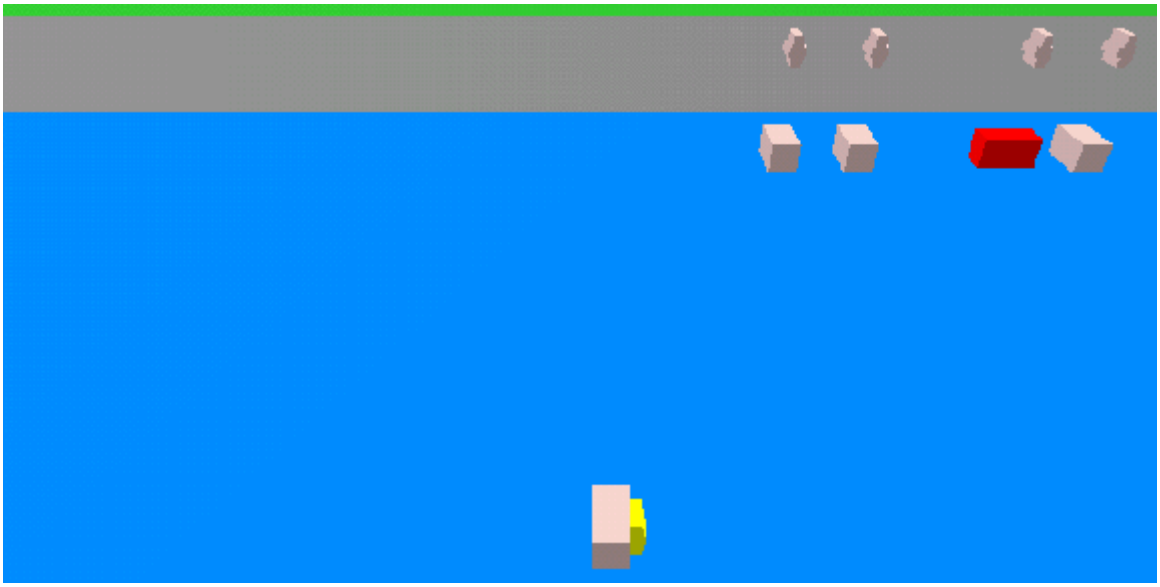
En este momento tenemos seleccionadas dos piezas, una del montaje y una unitaria. Se acoplara la pieza en la cara que tengan el mismo vector que la pieza unitaria seleccionada.

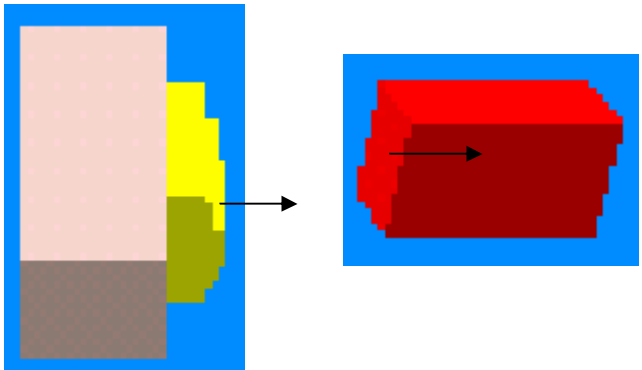


2. Pulsar la tecla P para realizar el montaje. Tendremos:



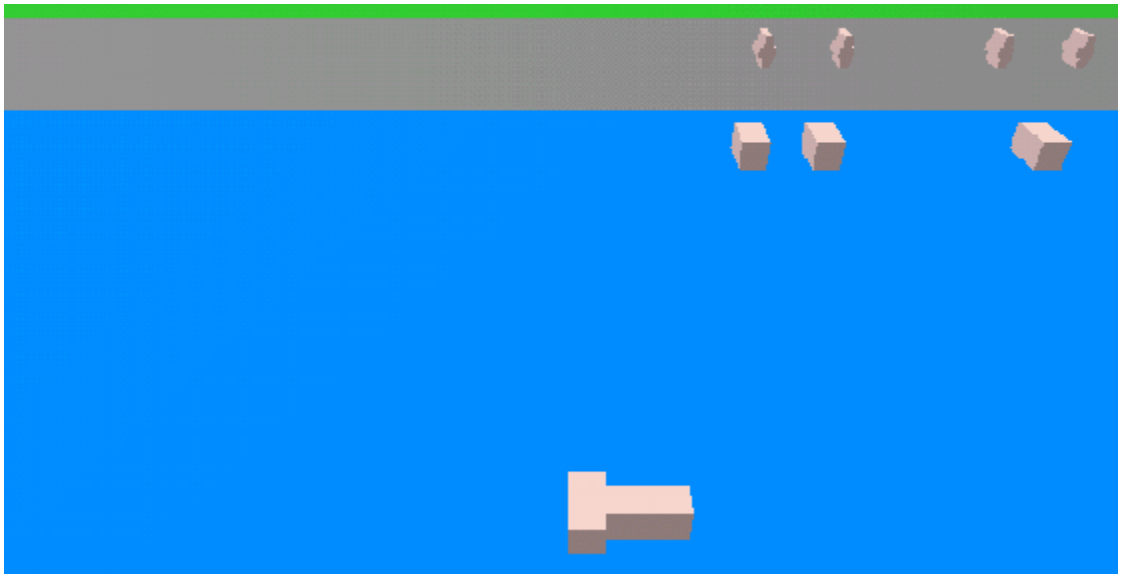
3. Vamos a volver al paso 1, seleccionar una pieza y decidiremos como la vamos a montar. Por ejemplo, escogeremos un prisma de 6 caras hembra y lo giraremos sobre Y 3 veces (270 grados). Escogeremos del montaje la pieza doble macho para montar. Tendremos:



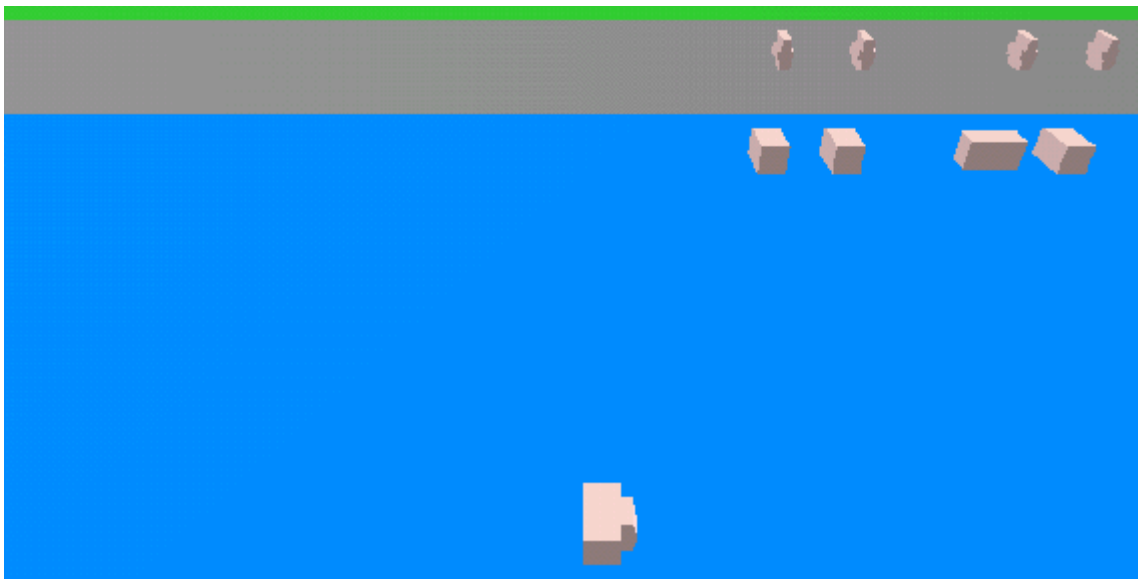


4. Al pulsar P conectaremos las dos piezas y la pieza roja pasará al montaje. Las piezas vuelven a su color inicial.

Siguiendo los pasos anteriores, esto es, seleccionando piezas del tipo deseado y con la posibilidad de poderse conectar sobre la pieza del montaje seleccionada por coincidencia del vector de la pieza con el vector de la cara, puede montarse el montaje deseado.



5. Si pulsamos la tecla N volvemos a tener las piezas en la situación anterior.

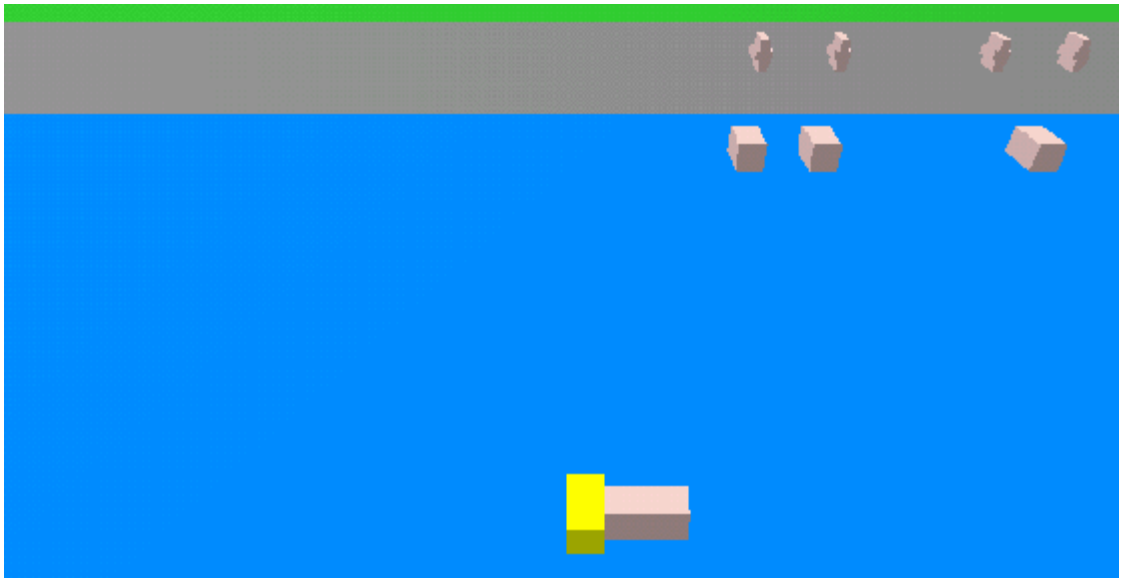


6. Si cuando tenemos seleccionada la pieza del Montaje (amarilla) pulsamos la tecla A seleccionamos otra pieza del Montaje según el orden en que las piezas fueron formando parte del Montaje.

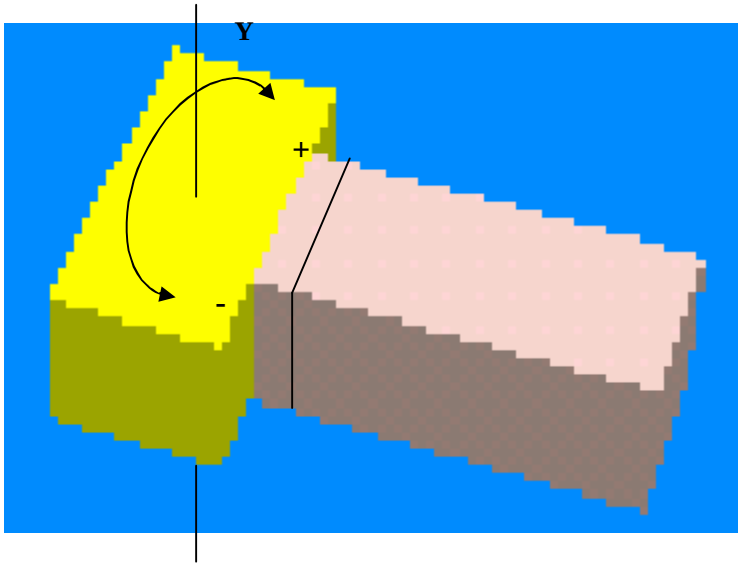
7. Si cuando tenemos seleccionada la pieza del Montaje (amarilla) pulsamos la tecla Q desseleccionamos la pieza pasando a su color inicial.

- Seleccionar pieza de Montaje. Aplicar estimulo a Montaje. Cambiar de pieza de Montaje. Desseleccionar Montaje.

1. Con el ratón se escogerá una pieza del Montaje sin tener seleccionada ninguna pieza unitaria o aislada. La pieza se coloreará de amarillo. Esta será la pieza a partir de la cual se aplicarán los estímulos al Montaje.

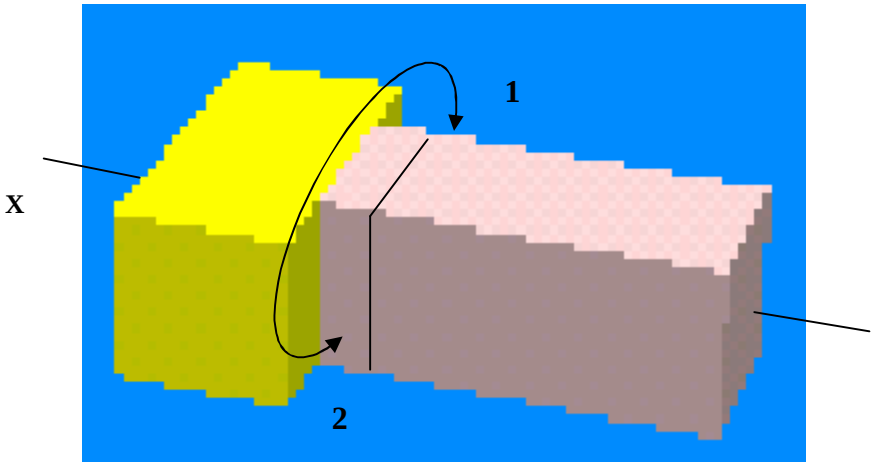


2. Primero aplicaremos estímulos con el teclado. Con las teclas + y - giraremos la pieza seleccionada 10 grados y esta arrastrará a las otras.

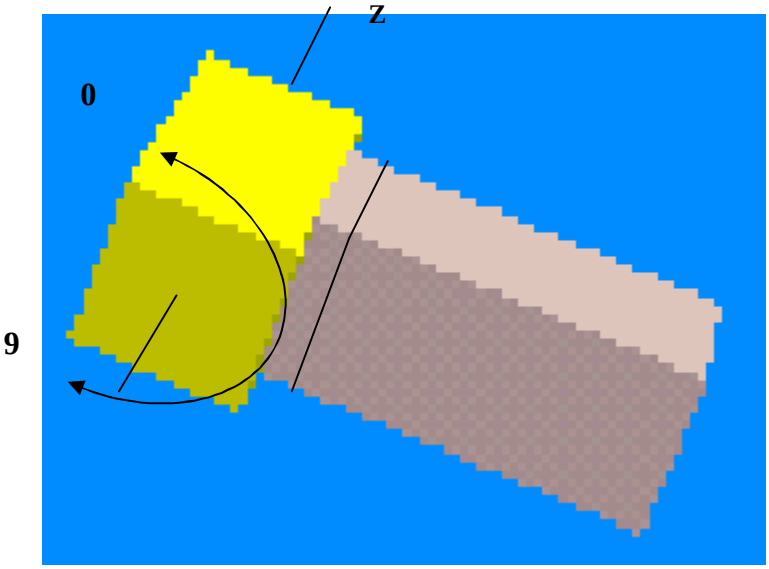


El proceso seguido es : se gira la pieza seleccionada, se busca en la matriz de conexiones del montaje caras ocupadas de dicha pieza y se detecta con que pieza esta conectada. Esta pieza se gira igualmente y a través de las caras de conexión se mueve. Idénticamente con la siguiente pieza.

Con las teclas 1 y 2 se gira 10 grados sobre el eje X.

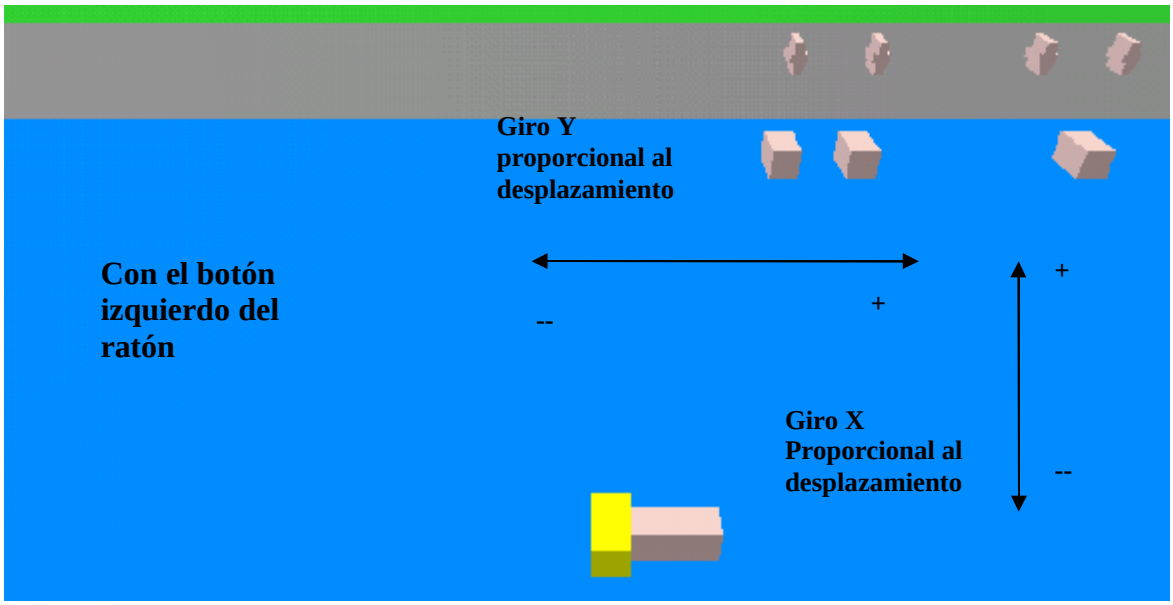


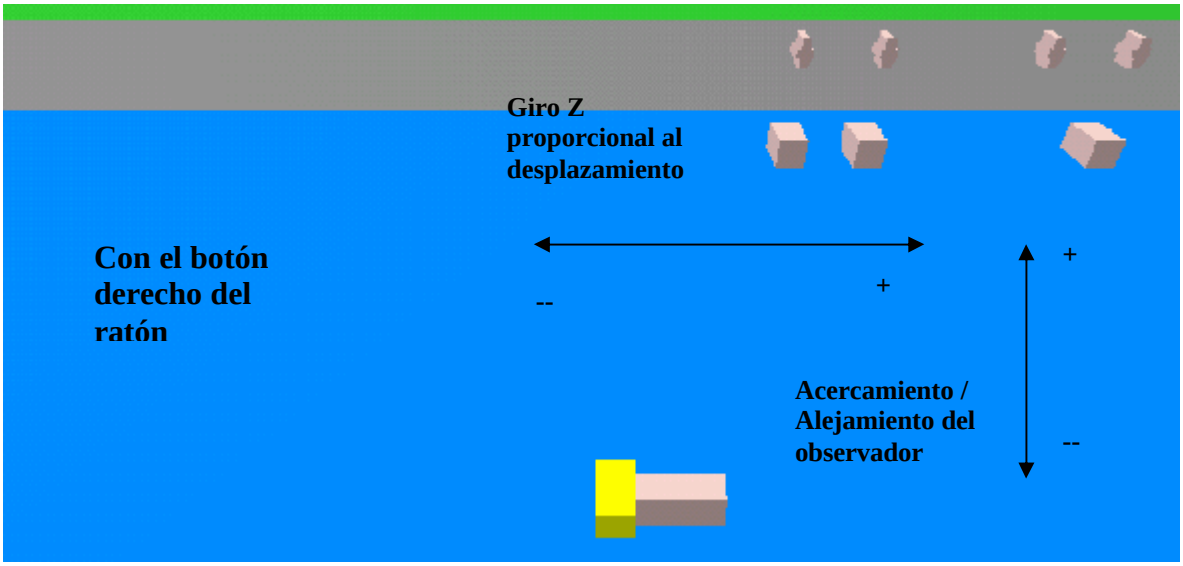
Con las teclas 9 y 0 se gira 10 grados sobre el eje Z.



Con las teclas de las flechas, así como con V y B podemos mover el Montaje hacia la derecha, izquierda, arriba, abajo, acercándose y alejándose del observador .

3. Ahora toca aplicar los estímulos por el ratón. Partimos del momento en que tenemos seleccionada la pieza del Montaje a la que vamos a aplicar los estímulos.





- Mover Observador.

  Cuando no hay ninguna pieza seleccionada podemos utilizar las teclas de las flechas y la V y B para mover el observador hacia la derecha, izquierda , arriba, abajo, acercarse o alejarse al centro de la escena, que permanecerá fijo.

- Simulador de proceso (Interprete).

  Al arrancar el programa pulsar la tecla D y después de confirmar los privilegios del NetScape se verá como una se secuencia de acciones se realizan por si solas. El proceso es el que describe el archivo datos.dat, que esta incluido en el anexo 2.

Observando el contenido del fichero y las acciones se entenderá que se puede conseguir con esta aplicación.

4.9. Comentarios adicionales

En la versión actual de la implementación no hay ninguna protección contra colisiones. Estas solo pueden ocurrir al mover piezas o montaje. El programa de montaje ve guiado por las cotas de los puntos a conectar en las caras hembra, donde conectan las cotas de los puntos de conexión de las caras macho.

5. Conclusiones

Personalmente creo que se ha conseguido sintetizar la esencia de un Mecano con piezas muy simples y operaciones básicas.

5.1. Logros

Los puntos fuertes que la aplicación cubre son:

- Se ha logrado una estructura de la información robusta.
- Se ha logrado una Interficie sencilla y completa.
- Se han cumplido las expectativas de los requisitos del proyecto.
- Se han superado las expectativas iniciales con el interprete y añadiendo la función deshacer último montaje.
- Se han añadido unas funciones, teclas 5 y 6, para ver la imagen como sólido y como líneas.
- Se pueden añadir fácilmente piezas con cualquier diseño y con conexiones hembra y se podría tener un Mecano de piezas estáticas, típico de los de construcción, como Lego o el mítico Exin Castillos.
- Se ha aprendido a gestionar el tiempo para realizar un proyecto, obteniendo una información y documentación de referencia para futuras aplicaciones.

5.2. Posibles mejoras

Evidentemente que se puede mejorar esta aplicación ya que realmente esta aplicación pretendía ser inicial. Estos puntos son una prueba:

- Añadir nuevas Piezas estáticas.
- Trabajar con varios montajes, o submontajes, organizados en vector.
- Añadir piezas de conexión y piezas dinámicas. (ver presentación)
- Mejorar el interfaz añadiendo cuadros de dialogo y de selección.
- Ampliar las funciones que se realicen con el ratón.
- Analizar que se puede mejorar moviendo el centro de la escena.

Bibliografía consultada

- Dirección de Internet donde he podido resolver el problema de firmar los archivos para tener los privilegios para poder guardar los archivos a disco duro.

  http://www.iec.csic.es/criptonomicon/java/nc.html

- Programas ejemplo de GL4Java, sobre todo cuando tuve que entender la selección de piezas clicadas por el ratón.
- Documentación de la asignatura de Informática Gráfica I.
- Documentación de la asignatura de Estructura de la Información.

Agradecimientos

Con este proyecto se cierra una etapa, consiguiendose unos objetivos personales muy deseados. Consecuentemente se abre otra etapa.

Es de bien nacido recordar a todos los que, por un motivo u otro, hemos recorrido algún tramo juntos durante esta etapa.

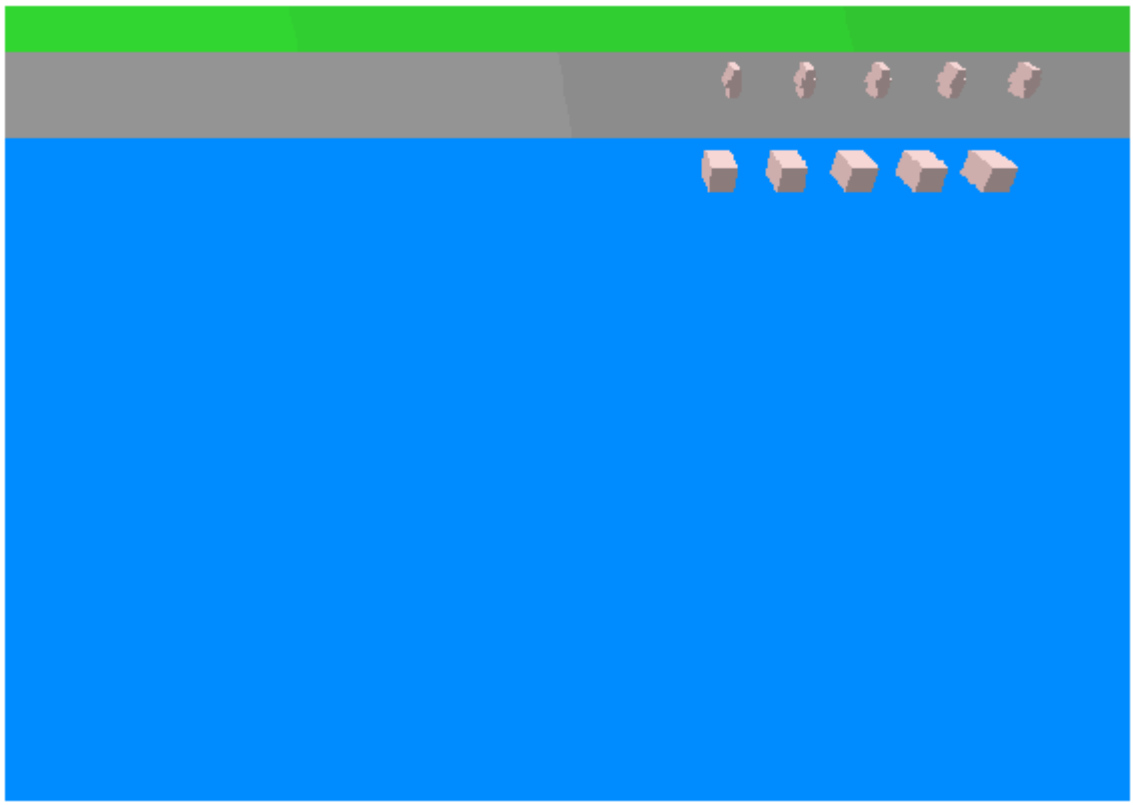
No quiero poner nombres, pero tanto compañeros, como tutores y por supuesto profesores han pasado por mi mente.

Pero si no llega a ser por mi familia ni lo hubiera empezado, ni lo hubiera continuado. Y eso que lo han sufrido con algunas fiestas o salidas de menos.

Gracias a todos.
Gracias a Loli, Susana y Tania.

Anexo 1. Ejemplo de uso.

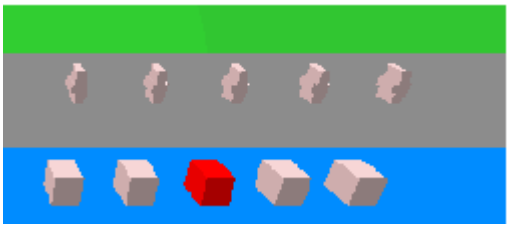
Pantalla gráfica al iniciar el programa



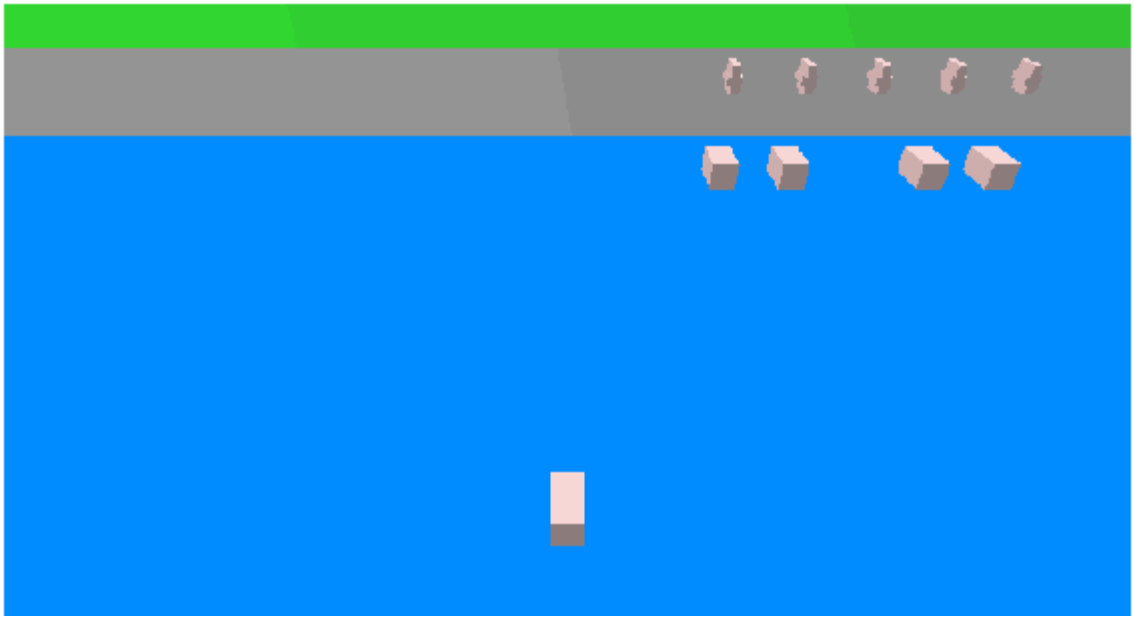
Información anexa para ayuda al uso (falta actualizar)

Inicializar Montaje	Montar diversas Piezas	Simular Montaje	Mover observador
Raton: Seleccionar Pieza	**1. Ratón**: Seleccionar Pieza	**1. Ratón**: Seleccionar Montaje	**Flechas**: Derecha/Izquierda
X,x: Girar Pieza eje X	**X,x,Y,y,Z,z**: Girar Pieza	**+, -**: Giro 10 grados eje Y	**Flechas**: Subir/Bajar
Y,y: Girar Pieza eje Y	**2. Ratón**: Seleccionar Montaje	**1,2**: Giro 10 grados eje X	**v,b**: Acercarse/alejarse
Z,z: Girar Pieza eje Z	**P,p**: Montar Pieza	**9,0**: Giro 10 grados eje Z	**Ficheros**
A,a: Cambio de Pieza	**N,n**: Deshacer ultimo montaje	**Flechas**: mover Montaje	**S,s**: Salvar posición actual
Q,q: Desseleccionar Pieza	**A,a**: Cambio de Pieza	**v,b**: acercarse o alejarse	**L,l**: Leer posición salvada
P,p: Iniciar Montaje	**Q,q**: Desseleccionar Pieza	**Q,q**: Desseleccionar	**I,i**: Situación de Inicio
		Drag izq X: Giro prograsivo eje Z	**D,d** Secuencia en fichero
		Drag izq Y: Giro progresivo eje X	
		Drag der X: Giro progresivo eje Y	
		Drag der Y: Acercarse/Alejarse	

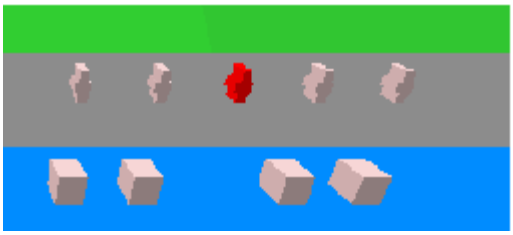
Área de las Piezas cuando hay una Pieza seleccionada
(Seleccionada con un click del ratón sobre la Pieza)



Pantalla gráfica cuando la Pieza seleccionada se ha pasado a Montaje
(se ha pulsado 'p')
(Piezas del Montaje en centro de la pantalla)



Selección de otra Pieza de distinto tipo que la anterior
(con un nuevo click sobre ella)



Selección de la pieza del Montaje donde se va a montar la Pieza seleccionada
(inicialmente es la primera)

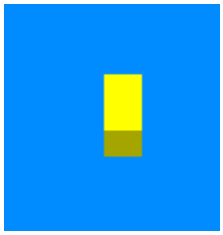
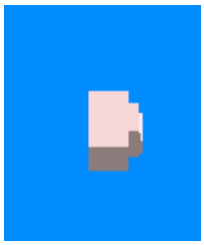
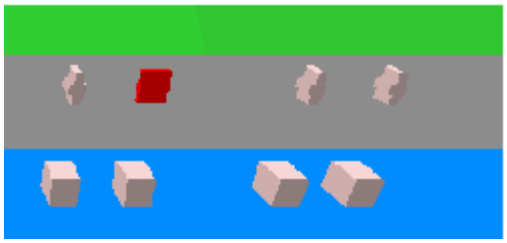


Imagen de Montaje con las Piezas montadas



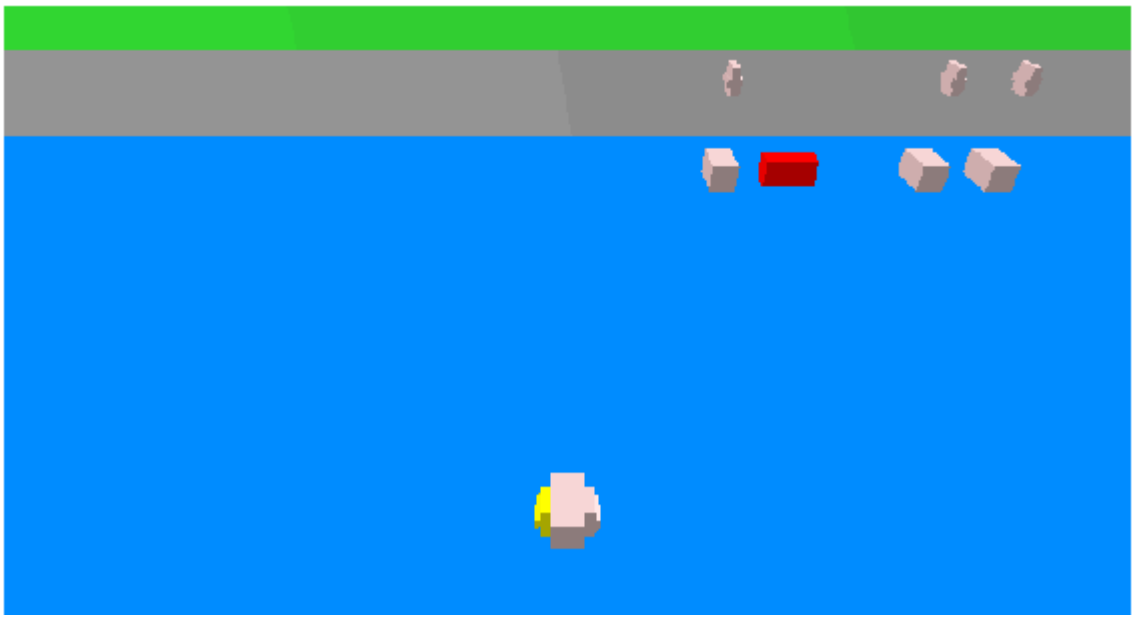
Selección de otra Pieza y girada en el eje Y 90 grados (tecla 'y')



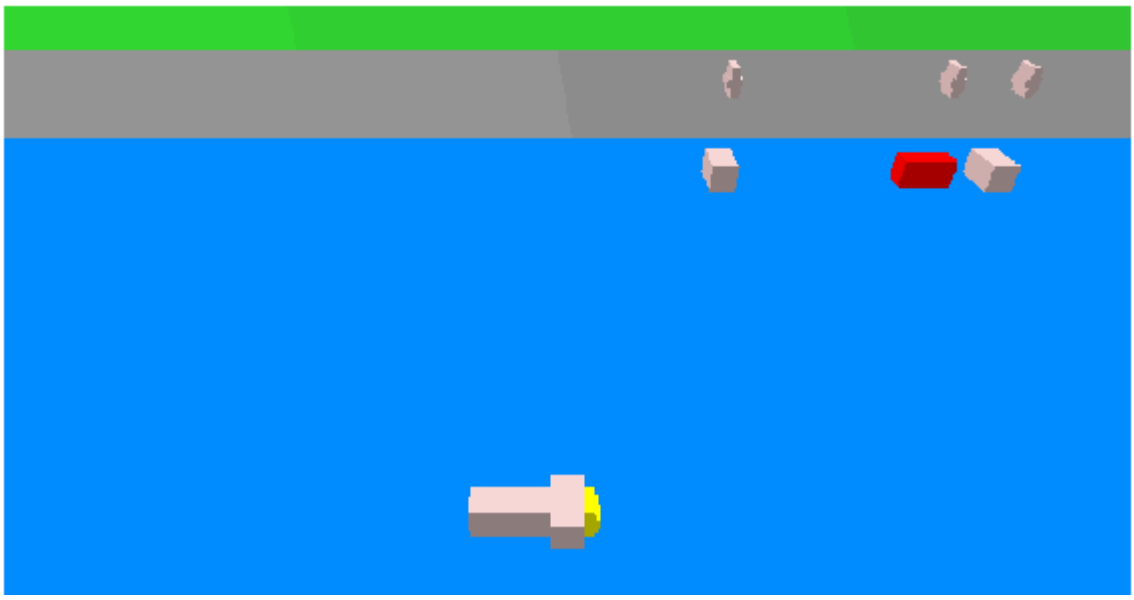
Vuelta a seleccionar la Pieza de referencia sobre el Montaje

Montaje con las tres Piezas (3ª pieza tiene 2 giros en Y, 180 grados)

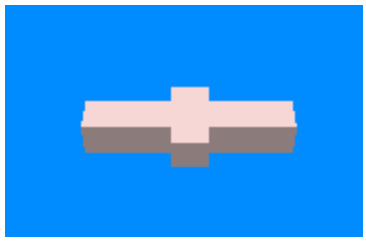
Selección de otra Piezas y nuevo giro de 90 grados en Y
Selección de Pieza del Montaje a unir



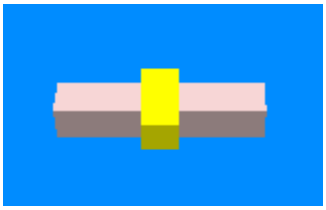
Resultado del Montaje
Nueva selección de Pieza con 3 giros en Y
Selección de Pieza del Montaje



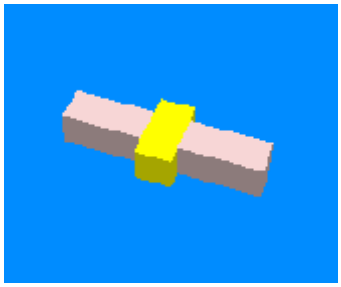
Resultado del Montaje



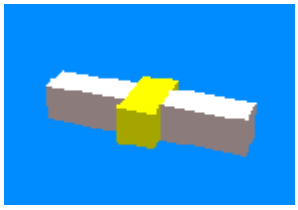
Selección de la Pieza a la cual se le va a dar el estímulo en la simulación
(entramos en estado de simulación)



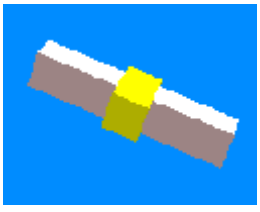
Giramos el Montaje sobre el eje Y (tecla '+' o '-')
(opcionalmente dragging del ratón con botón izquierdo y movimiento en X)



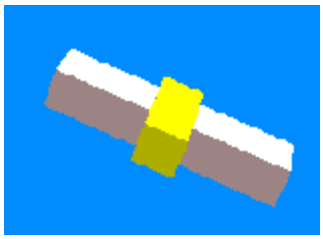
Girado el Montaje sobre el eje X (tecla '1' o '2')
(opcionalmente dragging del ratón con botón izquierdo y movimiento en Y)



Girado el Montaje sobre el eje Z
(opcionalmente dragging del ratón con botón derecho y movimiento en X)



Acercado el Montaje hacia el Observador
(opcionalmente dragging del ratón con botón derecho y movimiento en Y)



Anexo 2. Descripción del archivo de simulación.

Ejemplo de archivo de simulación. Simulacion.dat

Como se puede ver es bastante autodescriptivo

```
Selecciona Pieza
3                          // Número de la pieza
Añade a Montaje
Selecciona Pieza
7
Selecciona Montaje
3
Añade a Montaje
Selecciona Pieza
2
Rota Y                     // Rota 90 grados en el eje seleccionado
Rota Y
Rota Y
Selecciona Montaje
7
Añade a Montaje
Selecciona Pieza
6
Rota Y
Rota Y
Selecciona Montaje
3
Añade a Montaje
Selecciona Pieza
1
Rota Y
Selecciona Montaje
6
Añade a Montaje
Selecciona Montaje         // Entramos en simulación
3
Rota Montaje Y+                // Rota 10 grados en el sentido y eje
Rota Montaje Y+                // seleccionado
```

```
Rota Montaje Y+
Mueve Izquierda          // Mueve 2 mm en la dirección
Mueve Izquierda          // seleccionada
Mueve Izquierda
Mueve Arriva
Mueve Arriva
Mueve Arriva
Acerca
Acerca
Acerca
Rota Montaje X+
Rota Montaje X+
Rota Montaje X+
Rota Montaje Z+
Rota Montaje Z+
Rota Montaje Z+
Desselecciona
Fin                      // Termina la simulación dejando el
                         // Montaje como quede
```

Anexo 3. Descripción de los archivos de estado.

Ejemplo de archivo de simulación. inicio.dat o prueba.txt

Como se puede ver es bastante autodescriptivo. En rojo inicio de objeto, en azul inicio de atributo (solo primer elemento)

```
Montaje 0
[Numero de Piezas
Maximo]
10
[Numero de Caras Maximo]
6
[Numero de Piezas en
Montaje]
0
[Matriz de conexiones]
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
[Piezas Usadas]
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
Montaje 1
[Numero de Piezas
Maximo]
10
[Numero de Caras Maximo]
6
[Numero de Piezas en
Montaje]
0
[Matriz de conexiones]
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
[Piezas Usadas]
-1
-1
-1
-1
```

```
-1
-1
-1
-1
-1
-1
Pieza
0
[Tipo]
1
[Color]
0.7
0.6
0.6
[Profundidad]
3.0
[Centro de Gravedad]
50.0
95.0
55.0
[Vector Orientacion]
0.0
0.0
1.0
[Esta Montada]
false
[Cara Ocupada]
false
false
false
false
false
false
[Vertices]
49.25
94.25
53.5
49.25
95.75
53.5
50.75
95.75
53.5
50.75
94.25
53.5
49.25
94.25
56.5
49.25
95.75
56.5
50.75
95.75
56.5
50.75
94.25
56.5
[Normales de las Caras]
0.0
0.0
-1.0
0.0
0.0
1.0
-1.0
0.0
0.0
0.0
1.0
0.0
0.0
0.0
-1.0
0.0
[Caras de Conexion]
50.0
95.0
53.5
50.0
95.0
53.5
49.25
95.0
55.0
50.0
95.75
55.0
50.75
95.0
55.0
50.0
94.25
55.0
50.0
95.0
56.5
Pieza
1
( Pieza 1 a 4, al ser
del mismo, tipo no se
describen )
…
Pieza
5
[Tipo]
2
[Color]
0.7
0.6
0.6
[Profundidad]
1.5
[Centro de Gravedad]
50.0
100.0
55.0
[Vector Orientacion]
-1.0
0.0
0.0
[Esta Montada]
false
[Conexion Ocupada]
false
false
[Vertices]
49.6
99.75
54.25
49.6
100.25
54.25
49.8
100.15
54.25
49.8
100.75
54.25
50.2
100.75
54.25
50.2
100.15
54.25
50.4
100.25
54.25
50.4
99.75
54.25
50.2
99.85
54.25
```

```
50.2
99.25
54.25
49.8
99.25
54.25
49.8
99.85
54.25
49.6
99.75
55.75
49.6
100.25
55.75
49.8
100.15
55.75
49.8
100.75
55.75
50.2
100.75
55.75
50.2
100.15
55.75
50.4
100.25
55.75
50.4
99.75
55.75
50.2
99.85
55.75
50.2
99.25
55.75
```

```
49.8
99.25
55.75
49.8
99.85
55.75
[Normales de las Caras]
0.0
0.0
-1.0
-2.5096454E-6
0.0
1.0
-1.0
5.086263E-6
0.0
0.4472022
0.8944329
1.8196965E-6
-1.0
2.1192861E-6
0.0
0.0
1.0
3.1788923E-6
1.0
-2.1192861E-6
0.0
-0.44720903
0.8944295
1.8197244E-6
1.0
-5.086263E-6
0.0
-0.4472022
-0.8944329
1.8196965E-6
1.0
-2.1192861E-6
```

```
0.0
0.0
-1.0
3.1788923E-6
-1.0
2.1192861E-6
0.0
0.44720903
-0.8944295
1.8197244E-6
[Puntos de Conexion]
50.2
100.0
55.0
49.8
100.0
55.0
Pieza
6
( Pieza 6 a 9, al ser
del mismo, tipo no se
describen )
```