ESCUELA SUPERIOR DE INGENIERÍA

INGENIERO TÉCNICO INFORMÁTICA DE GESTIÓN

DESARROLLO DE UNA APLICACIÓN INFORMÁTICA EN TIEMPO REAL CON ENTRADA G-CODE PARA MÁQUINA DE CONTROL NUMÉRICO (CNC) EN 3D.

Autor del Proyecto: José Luis Aparicio Rodríguez

Director del Proyecto: Arturo Morgado Estévez

Departamento: Ingeniería de sistemas y automática, tecnología electrónica y electrónica

Cádiz, Febrero 2011

Fdo: José Luis Aparicio Rodríguez

Agradecimientos

A mi familia por su apoyo en todo momento, a mis amigos por estar siempre dispuestos a ayudarme, y a mi tutor de proyecto por su ayuda y colaboración en este proyecto fin de carrera.

Sin vosotros no habría sido posible llevar a cabo este proyecto. En reconocimiento a todos ellos, seguiré siempre adelante y conseguiré todas las metas propuestas.

Por todo esto y más, Gracias.

Licencia

Índice General

Índice de Figuras

Capítulo 1

Introducción

En estos últimos años hemos sufrido grandes cambios tecnológicos, estos avances han aportado al ser humano grandes beneficios, desde la invención de aparatos, dispositivos y aplicaciones, hasta la creación y mejora de herramientas.

Estos desarrollos y cambios tecnológicos han provocado que haya un aumento en la automatización de procesos.

La automatización de procesos ha producido una enorme disminución de los costes temporales, mejorando por tanto la eficiencia de la producción, así como una mejora sustancial en el resultado final del producto, ya que disminuyen considerablemente los defectos de fabricación.

Ciñéndonos al mecanizado de piezas, debemos decir que gracias al desarrollo de maquinas-herramientas tales como máquinas de control numérico por computadoras, o CNC, podemos crear piezas de un muy alto grado de complejidad, con gran precisión y en poco tiempo, a diferencia de lo que pudiera hacer un operario manualmente, lo que seria mas que complicado.

Este proyecto trata de crear una aplicación para poder controlar una maquina CNC, además de dirigir el posicionamiento y actuación de la maquina a través de un programa de torneado definido.

1.1. Objetivos

El objetivo de este proyecto no es otro que el de poder interactuar con una maquina de control numérico o CNC.

Las formas de interactuar de nuestro proyecto es:

- Controlar en tiempo real el movimiento de la máquina.
- Representar gráficamente la secuencias de instrucciones con códigos G, alojados en un fichero.

- Actuar sobre la maquina siguiendo secuencias de instrucciones de un programa previamente creado.

1.2. Antecedentes

En el mercado actual existen varios programas comerciales y libres que tratan el mismo tema abarcado por nuestro proyecto

Varios de los ejemplos mas significativos son:

- **MACH:** Tal vez sea el programa comercial sobre control y simulación cnc mas importante. Mach tiene diferentes versiones : Mach1, Mach2 y Mach3, fue creado por la compañía Artsoft y funciona bajo windows.
- **EMC:** Es el programa libre, bajo licencia GNU más importante, funciona sólo bajo linux.
- **CNCPRO:** Es un programa libre, que trabaja bajo MsDOS.

1.3. Estudios previos

Los estudios previos que se llevaron a cabo para la realización de este proyecto fueron los siguientes:

Máquina de control numérico Para realizar este proyecto lo primero que se tuvo que estudiar fue en qué consistía, para qué servía y cómo funcionaba una máquina de control numérico. Gracias al tutor del proyecto y a compañeros que habían tratado previamente con la máquina de control numérico, aprender todo lo relacionado con este tipo de máquina-herramienta se hizo más llevadero.

Códigos G Para llevar a cabo este proyecto tuve que informarme acerca de todo lo relacionado con la programación cnc mediante códigos G estándar, y los distintos comandos utilizados para representar todo tipo de movimientos y figuras posibles mediante este sistema de programación. Para conocer mejor todo lo relacionado con códigos G, tuve la ayuda de mi tutor y de compañeros que conocían previamente este tema. Además de ello encontré libros dedicados a este tema que me resultaron muy útiles.

Qt y Qt Creator Este proyecto está íntegramente desarrollado mediante las bibliotecas Qt y en el entorno de desarrollo integrado Qt Creator, Qt se basa principalmente en el lenguaje de programación C++, para conocer bien todo lo relacionado con las bibliotecas Qt, tuve la gran ayuda de su web principal donde existe numerosa documentación en relación a sus clases y librerías, además de tener gran cantidad de ejemplos que me resultaron muy útiles.

OpenGL El apartado de representación gráfica esta basado en OpenGL, que es un conjunto de librerías dedicada a la elaboración de gráficos en 2 y 3 dimensiones, como es el caso. Para conocer mejor todo lo relacionado con estas librerías, encontré varios libros en la biblioteca dedicados a este tema, además de contar con una amplia documentación online de OpenGl.

Comunicación serie En un primer momento se desarrolló la aplicación orientada a la comunicación serie con la máquina de control numérico por lo que tuve que ampliar mis escasos conocimientos acerca de los protocolos de comunicación serie, principalmente busqué información sobre QextSerialPort que servía para la comunicación serie en windows bajo Qt.

Comunicación paralela La aplicación final usa comunicación por puerto paralelo para comunicarse con la máquina de control numérico por lo que para conocer mejor este tipo de comunicación tuve que buscar información acerca de los protocolos de comunicación por puerto paralelo y cómo implementarlo en windows y bajo c++.

Latex Para realizar la documentación del proyecto tuve que buscar información acerca de programación en Latex, para ello encontré libros en la biblioteca dedicados a ello, además de esto, encontré varios manuales y tutoriales por la web que me sirvieron de gran ayuda.

1.4. Equipo de desarrollo

Las herramientas utilizadas en este proyecto tanto a nivel hardware como software son las siguientes:

1.4.1. Hardware empleado

- PC empleado para el desarrollo del software.
- PC del laboratorio donde se realizaron las pruebas.
- Maquina CNC Sable 2015.
- Analizador Lógico DigiView 3100.

Marca	Toshiba Satellite
Procesador	Intel Centrino Duo 1.83GHz
Memoria RAM	1 GB
Disco Duro	100 GB
Tarjeta Vídeo	NVIDIA GeForce 7300

Figura 1.1: Hardware PC empleado para el desarrollo del software

1.4.1.1. Máquina de control numérico CNC Sable 2015

Se considera de Control Numérico por Computador, también llamado CNC , a todo dispositivo capaz de dirigir el posicionamiento de un órgano mecánico móvil mediante órdenes elaboradas de forma totalmente automática a partir de informaciones numéricas en tiempo real.

Entre las operaciones de maquinado que se pueden realizar en una máquina CNC se encuentran las de torneado y de fresado. Sobre la base de esta combinación es posible generar la mayoría de las piezas de industria.

Este es, sin duda, uno de los sistemas que ha revolucionado la fabricación de todo tipo de objetos, tanto en la industria metalúrgica como en muchos otros ámbitos productivos.

Para mecanizar una pieza se usa un sistema de coordenadas que especificarán el movimiento de la herramienta de corte.El sistema se basa en el control de los movimientos de la herramienta de trabajo con relación a los ejes de coordenadas de la máquina, usando un programa informático ejecutado por un ordenador.

Para la automatización de rutinas de ejecución de la máquina se usan una serie de datos y códigos que describen las acciones a realizar por parte de la máquina de control numérico. Estos programas que contienen estos códigos son códigos Gs regidos bajo la norma DIN 66024 y 66025. En el Apendice A se describen los códigos utilizados.

Con respecto a la máquina que nos ocupa : CNC Sable 2015 decir que tiene las siguientes características.

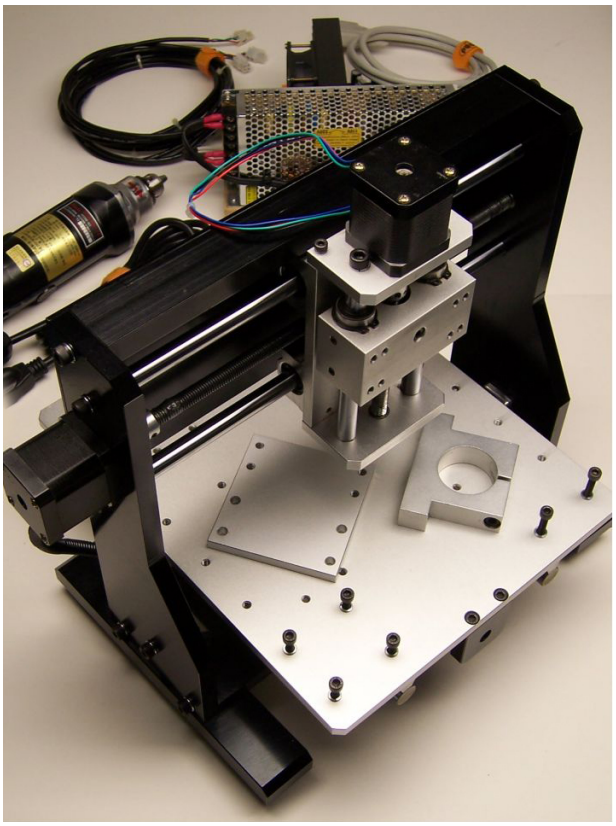

Figura 1.2: CNC Sable 2015

Cuerpo:

1. Ejes: X-200mm Y-Z-150mm 60mm
2. Tamaño de la tabla: 250mm x 310mm
3. Espesor máximo del material: 60 mm
4. Leadscrews: M10 de tono = 1,5 mm
5. Material: Construida en aleación de aluminio (anodizado)

Motor paso a paso:

1. Ángulo de paso: 1.8 grados
2. Número de Fases: 2
3. Hold Torque: 4.4 kg.cm

4. Corriente nominal / Fase: 1.68 amperios.
5. Resistencia de fase: 1.65 ohmios
6. Voltaje / Fase: 2.77 VDC

1.4.1.2. Analizador Lógico DigiView 3100

Un analizador lógico es un instrumento de medida que captura los datos de un circuito digital y los muestra para su posterior análisis, de modo similar a como lo hace un osciloscopio, pero a diferencia de este, es capaz de visualizar las señales de múltiples canales. Además de permitir visualizar los datos para así verificar el correcto funcionamiento del sistema digital, puede medir tiempos entre cambios de nivel, número de estados lógicos, etc. La forma de capturar datos desde un analizador lógico es conectando una punta lógica apropiada en el bus de datos a medir.

Los analizadores son empleados principalmente para la detección de errores y comprobación de prototipos antes de su fabricación, comprobando las entradas y analizando posteriormente el comportamiento de sus salidas.

El analizador lógico utilizado para las pruebas de comunicación entre la aplicación y la máquina CNC fue el analizador DigiView 3100.

Figura 1.3: Analizador lógico DigiView 3100

1.4.2. Software empleado

Sistema Operativo	Windows XP
IDE	QT Creator
Lenguaje de programación	C++
Generación de documentos	LyX
Creación de diagramas	DIA
Creación diagrama Gantt	GanttProject
Gráficos, iconos, imágenes	GIMP 2
Analizador Lógico	Digi View Capture & Analysis Software

Figura 1.4: Software empleado

Capítulo 2

Desarrollo del Calendario

2.1. Descomposición de las actividades

En este proyecto se ha decidido utilizar el modelo incremental en el apartado metodología de desarrollo se especifican con más claridad las razones de esta decisión.

El modelo incremental se compone de incrementos.

2.1.1. Incremento 1: Acercamiento al proyecto

El 26 Abril de 2010 mi tutor de proyecto me asigna el proyecto de realización de una aplicación para controlar una máquina de control numérico. Ese mismo dia me muestran la máquina cnc del laboratorio para ver en que consiste y cómo actúa. Además de ello se me comenta cuales serían los pasos iniciales que tendría que dar para comenzar.

En esta fase del proyecto se barajaron las diferentes opciones de implementación tales como: Visual c++, Borland, VB, Qt etc... , la decisión final fue la de desarrollar la aplicación con Qt, mediante la IDE Qt creator, ya que Qt trabaja en c++ y además es software libre.

A partir de ese momento me dispuse a buscar información acerca de Qt.

En este primer incremento se desarrolló la aplicación inicial. Consistía en una aplicación para una cnc de 2 ejes. Su interfaz constaba de:

- Flechas para 4 direcciones.
- Velocidad de movimiento x1 x2 x5 x10 x100.
- Coordenadas relativas.
- Coordenadas Absolutas.
- Reset x, y coordenadas relativas.

- Visor Pantalla movimiento.
- Información acerca de los limites de los ejes x e y.

Con respecto a su implementación teníamos:

- Clase CNC, donde se desarrollaron los siguientes métodos:
  - MoverUp()
  - MoverDown()
  - MoverRight()
  - MoverLeft()
  - ResetX()
  - ResetY()
  - velx1()
  - velx2()
  - velx5()
  - velx10()
  - velx100()
  - Utilización de keyPressEvent para mover con teclado

Al pulsar sobre las flechas de dirección o pulsar las teclas de dirección se incrementa o decrementa las coordenadas absolutas o relativas, y se mueve un puntero en pantalla, que representa el movimiento de la maquina en tiempo real. Está primera versión fue finalizada el 29 de abril de 2010.

Aquí tenemos una impresión de pantalla de nuestro primer programa.

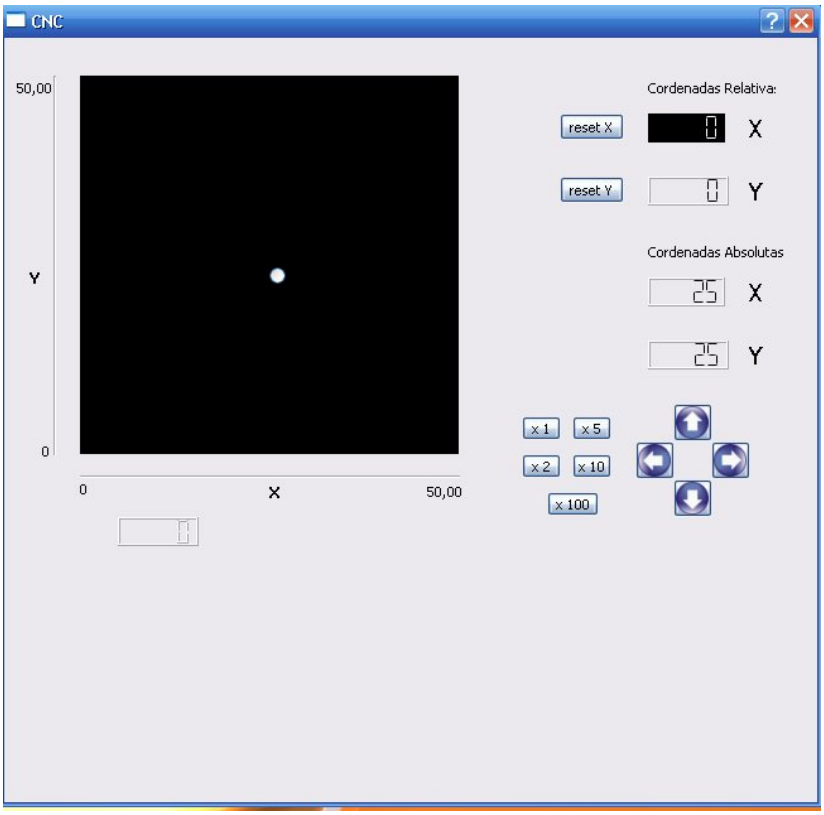

Figura 2.1: CNC v0

2.1.2. Incremento 2: Dibujar lineas

Antes de desarrollar este segundo incremento tuve una reunión con el tutor el 30 de abril de 2010 donde le mostré el incremento 1 y me comentó las posibles mejoras y cambios que tenía que realizar en el segundo incremento.

Para la realización de la segunda versión seguí buscando información acerca de las máquinas de control numérico, para tener un conocimiento general de su finalidad. También traté de mejorar mi conocimiento sobre Qt, y Qt creator. Más concretamente hice más hincapié en la búsqueda información sobre pintar lineas.

La segunda versión del producto tenia las siguientes características:

- Con respecto a su interfaz:
  - Posibilidad de pintar lineas y cerrar el polígono, pudiendo resetear la ultima linea y borrar el dibujo completo según se desee.
- Con respecto a su implementación

- Se modificó la clase Cnc creando las siguientes funciones
- Utilización paintEvent()
- PintarLinea()
- ResetLinea()
- CerrarPoligono()
- BorrarDibujo()

En esta segunda versión ya podíamos dibujar lineas. Esta segunda versión fue finalizada el 4 de mayo de 2010.

La segunda versión del programa tenia el siguiente aspecto:

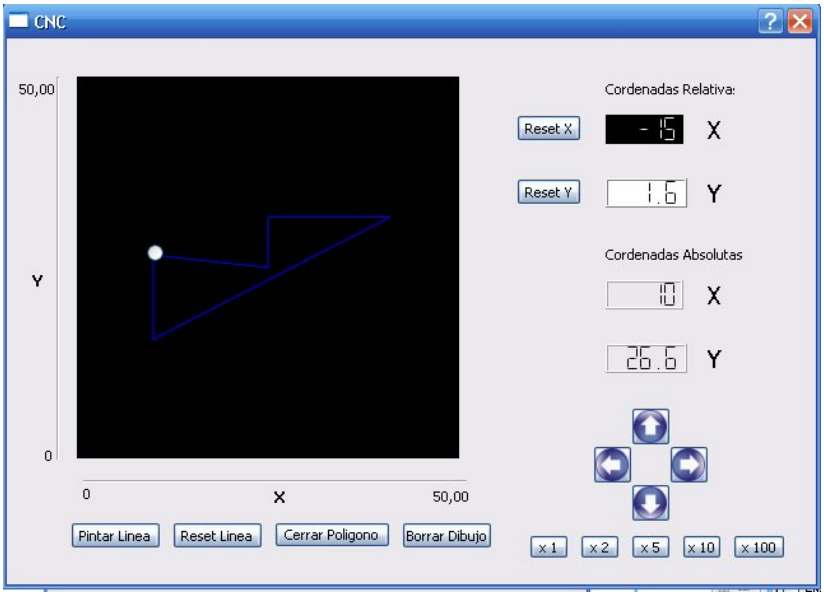

Figura 2.2: CNC v1

2.1.3. Incremento 3 : Dibujar Polígonos

Antes de la realización de esta tercera versión de la aplicación tuve una reunión con mi tutor el 6 de mayo de 2010 donde le mostré la segunda versión y me tutorizó para la realización de este tercer incremento.

Los cambios mas significativos en cuanto a la interfaz fueron los siguientes:

- Interfaz
  - Diferenciación entre 2 modos de ejecución:
    - “Modo Dibujar” y "Modo Control".

  - Posibilidad de dibujar más de un polígono en la pantalla, pudiéndose eliminar el ultimo polígono dibujado.

- Implementación

  - Modificación paintEvent() y creación de los siguientes métodos:
  - ModoDibujar()
  - SalirModo()
  - NuevoPoligono()
  - BorrarPoligono()
  - Creación de la clase Pintar

La finalización de esta tercera versión se finalizó el 16 de mayo de 2010 y tenía el siguiente aspecto:

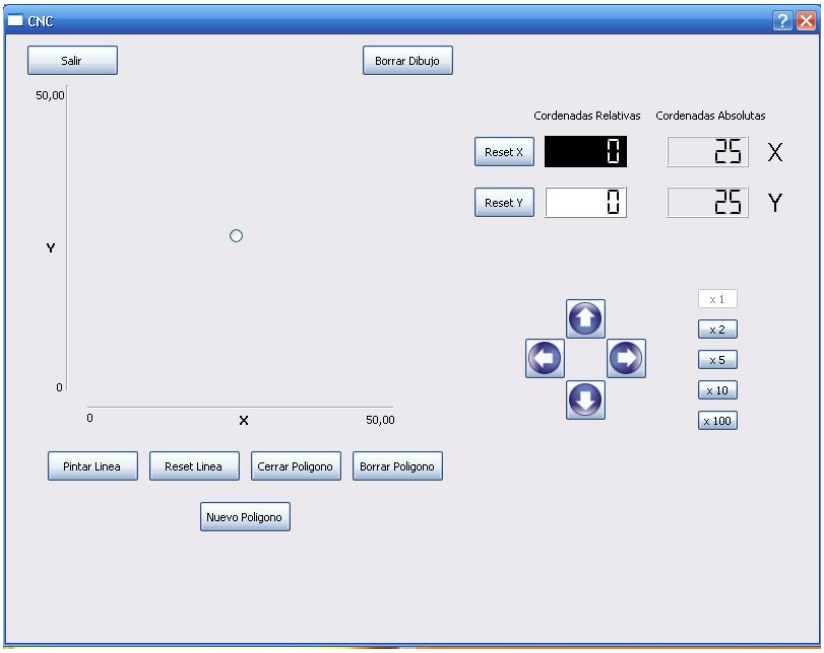

Figura 2.3: CNC v2

2.1.4. Incremento 4 : Comunicación serie

Antes de la realización de este cuarto incremento tuve una reunión con mi tutor el 14 de junio de 2010 en la que le mostré los avances realizados en la tercera versión del programa y me comento los cambios y mejoras que tenia que realizar para este cuarto incremento.

Uno de estos cambios era incluirle comunicación serie, ya que en un principio se creía que la máquina podría comunicarse de ese modo con la aplicación.

Por ello me dispuse a buscar información acerca de la comunicación por puerto serie, también realicé una búsqueda de información acerca de Qextserialport para implementar la comunicación serie en Qt.

Los cambios producidos en esta 4ª versión fueron los siguientes

- Cambios en la interfaz
  - Inclusión del eje z y posibilidad de resetearlo, no incluye botón de direccionamiento z.
  - Inclusión primer acercamiento de comunicación con la máquina, la comunicación se realizaba mediante puerto serie especificando el puerto COM al que queríamos conectarnos.
- Cambios en la implementación
  - Eliminación clase Pintar
  - Utilización Qextserialport
  - Creación de la función ResetZ()
  - Conectar()
  - Desconectar()
  - Creación de la clase Comunicación y los siguientes métodos
    - Comunicación(QString)
    - trasmitir(const char*)
    - QextSerialPort* Puerto()
    - bool IsOpen()
  - Creamos Clase PortListener con el método receive() para recibir información por puerto serie.
- Pruebas
  - Se realizan multitud de pruebas hasta conseguir comunicación por puerto serie, el problema que teníamos era que configurábamos el puerto antes de abrirlo, en lugar de hacerlo al contrario.

  El 6 de julio se consiguió la comunicación con la placa por medio del puerto usb, las pruebas se realizaron con un prototipo de movimiento de cámara por puerto serie, desarrollado por un compañero.

El aspecto de esta versión era la siguiente:

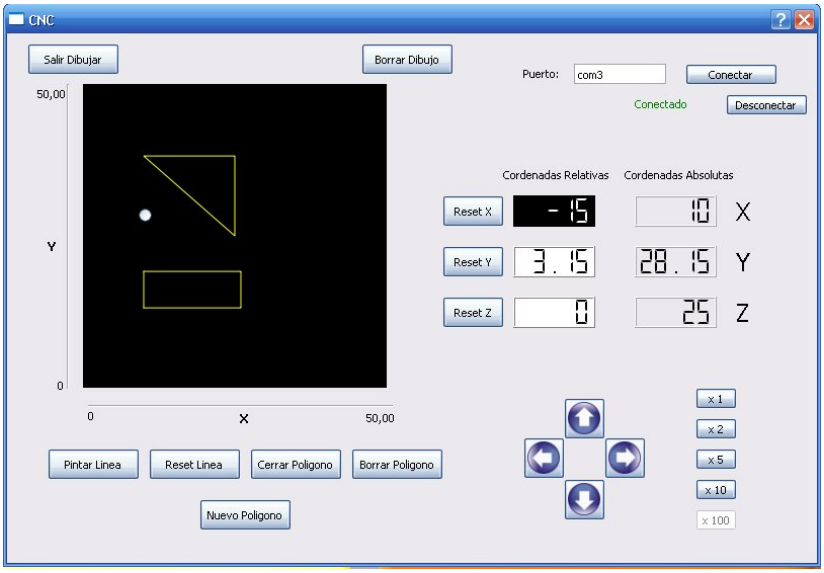

Figura 2.4: CNC v3

2.1.5. Incremento 5 : Intento de representar 3d

En esta versión se intentó realizar una representación de las 3 dimensiones en nuestra aplicación esto se hizo con la inclusión de 4 pantallas para representar las distintas perspectivas en 3 dimensiones, sólo se movía y desarrollamos una de ellas.

Este era su aspecto:

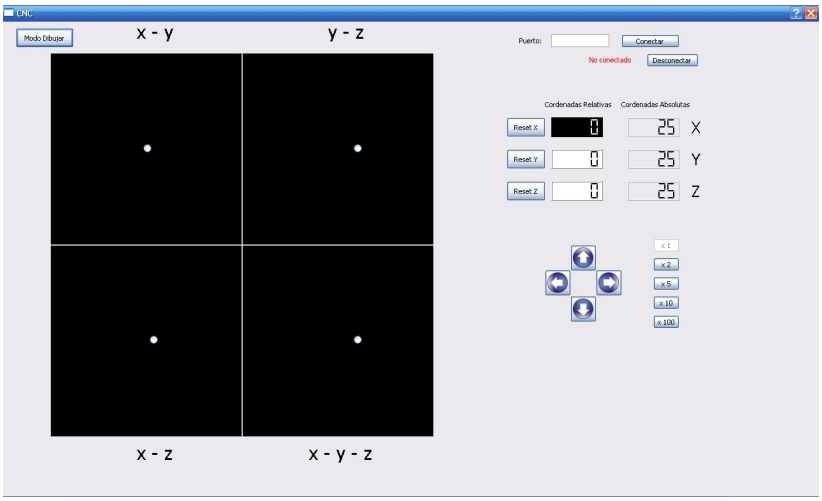

Figura 2.5: CNC v4

2.1.6. Incremento 6 : Ficheros cnc

Antes de realizar esta sexta versión tuve una reunión con mi tutor de proyecto en la que le mostré los avances que había hecho con la aplicación. En esa reunión me comentó la posibilidad de incluirle procesamiento de códigos Gs. Por lo que me puse a realizar una recogida de información acerca de Códigos G, además de realizar una búsqueda de información sobre manejo de ficheros en Qt, y una búsqueda de información acerca Vcarve y la creación de ficheros cnc mediante comandos Gs.

En este sexto incremento se realizaron numerosos cambios y mejoraras del producto, tales como:

- Con respecto a la interfaz
  - Se añade 2 botón para controlar el eje z, se ve reflejado en el incremento o decremento de las coordenadas absolutas y relativas.
  - Se añade la posibilidad de cargar un fichero cnc con códigos Gs, se filtra y se muestran tanto el original como el filtrado, mediante 2 visores de ficheros.
  - Se puede cerrar el fichero.
  - Se añade botón RUN para futura ejecución, (implementado pero aun no funciona correctamente).
  - Se representa el fichero en el visor gráfico, aunque no representa arcos, solo lineas , todo esto en 2 dimensiones.
- Con respecto a la implementación se hicieron los siguientes cambios:

- Modificación Clase PortListener donde se vio simplificada
- Se añade las funciones Activador()
- MoverZUp()
- MoverZDown()
- AbrirFichero()
- CerrarFichero()
- MostrarFichero()
- Run()
- Creación clase Fichero y el método esencial Procesar()
- Se crea la clase Parar que utiliza msleep para llevar el control de velocidad, aunque se deshecho mas adelante.

- Con respecto a las pruebas, se realizaron varias pruebas con varios ficheros generados por programas externos y por mi, lo que produjo un perfeccionamiento en el procesado y representado del fichero.
- Este sexto incremento se finalizó el 20 de julio de 2010 y tenia la siguiente interfaz:

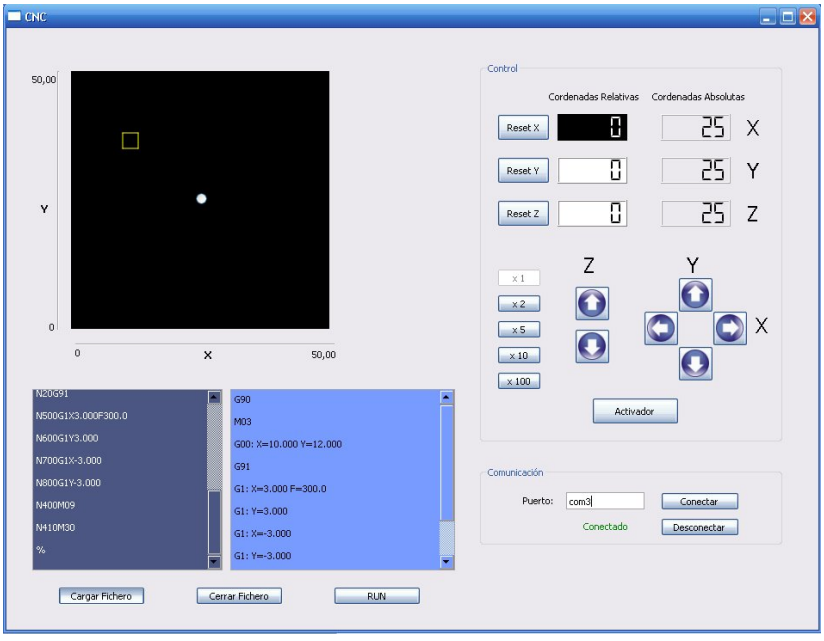

Figura 2.6: CNC v5

2.1.7. Incremento 7 : Visor gráfico 3d

Antes de la realización de este séptimo incremento tuve una reunión con mi tutor de proyecto en la que le mostré el avance realizado en el proyecto y me propuso mejoras, en esta reunión además se estuvo barajando las diferentes máquinas cnc que se podían adquirir, al final tomamos la decisión de adquirir la máquina de control numérico sable cnc 2015.

En este incremento también se llevo a cabo una labor de documentación de todos los avances realizados en el proyecto hasta el momento, para llevar a cabo la documentación se buscó información acerca de Latex y de como se podría implementar, llegando a la conclusión de utilizar LyX como procesador de texto en Latex.

Con respecto al programa en si, se desarrollo un visor gráfico en 3 dimensiones, para llevar a cabo esto tuvimos que elegir el método de implementación, que finalmente fue OpenGl por su alta compatibilidad con Qt y por ser de libre distribución. Por todo esto tuvimos que buscar información acerca de su implementación y su uso en esta plataforma.

Los cambios producidos en la séptima versión del producto tanto a nivel de interfaz como de implementación fueron los siguientes:

- Interfaz
  - Se añade botón ayuda, para mostrarla en un futuro
  - Se modifica el botón de ejecución , abrir fichero ,y cerrar fichero mostrándolos con iconos gráficos
  - Se pueden representar arcos en el visor gráfico del fichero
  - Se representan figuras en 3 dimensiones pudiéndose mover con el ratón para su mejor visión.
- Implementación
  - Creación clase Gráfico y el método principal draw() utilizando OpenGL.

La séptima versión del producto se finalizó el 24 de septiembre de 2010 y tenia el siguiente aspecto:

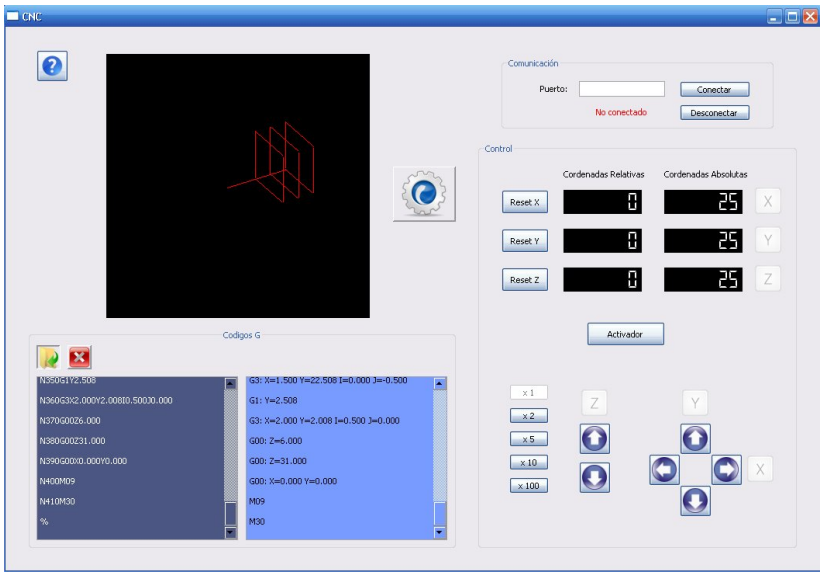

Figura 2.7: CNC v6

2.1.8. Incremento 8 : Ejecución

Antes de realizar este incremento tuve 2 reuniones con mi tutor de proyecto el 15 de octubre y el 3 de noviembre, en las que en un primer momento buscamos el tipo de información que se envían a las máquinas de control numérico, para ello se utilizó el analizador lógico DigiView con el software Mach2 y el hardware Cenece5.

Más adelante me dispuse a buscar información acerca de la comunicación usb-paralela, para ello busqué información acerca de la comunicación paralela creyendo que se utilizaba de igual modo , también seguí utilizando el analizador lógico para realizar pruebas de comunicación paralela.

En cuanto a los cambios producidos por la aplicación en este incremento, fueron los siguientes:

- A nivel de interfaz
  - Se añade el botón acerca de... donde se muestra la información acerca de la aplicación
  - Se muestra el manual en un navegador al pulsar sobre tal botón, aunque aun no muestra el manual, solo hola mundo
  - Posibilidad de ejecutar la aplicación aunque sin máquina aun donde ejecutarla y probarla.

- En la implementación se vieron los siguientes cambios
  - Creación de los métodos
    - Ayuda()
    - Manual()

Esta octava versión se finalizó el 21 de noviembre de 2010 y constaba de la siguiente interfaz:

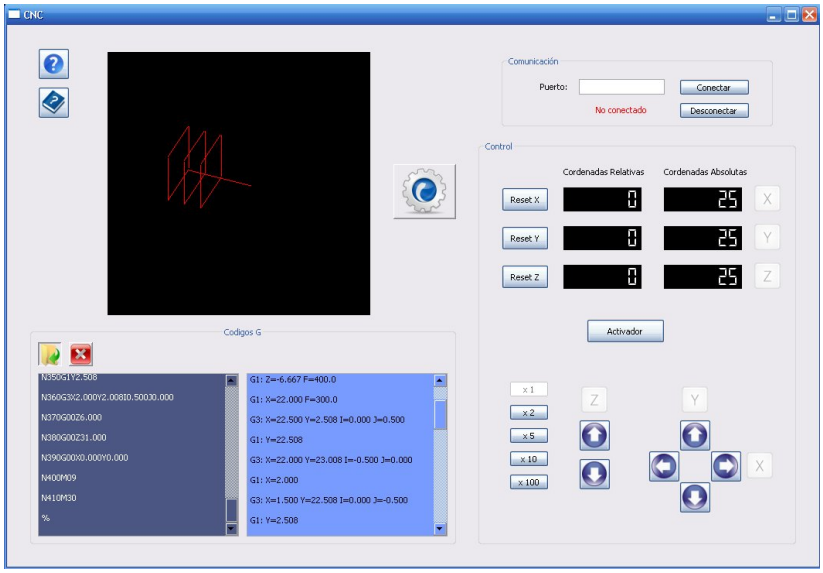

Figura 2.8: CNC v7

2.1.9. Incremento 9 : Programa puerto paralelo

Antes de realizar este noveno incremento tuve una reunión con mi tutor el 22 de noviembre de 2010 en la que me comentó la llegada de la máquina de control numérico Sable 2015 (21/11/2010). Ese mismo dia se intentó calibrarla y ponerla en funcionamiento.

Se plantea la opción de crear un programa para comunicarse por el puerto paralelo. Para llevar a cabo este cometido, me dispuse a buscar información acerca de la comunicación con el puerto paralelo, y como realizar esto en Qt.

Llegué a la conclusión de utilizar la librería inpout32.dll, que es una librería de libre distribución y que sirver para poder comunicarse con el puerto paralelo en el sistema operativo windows xp, más adelante se relataran las características de esta librería y el por qué de su utilización.

En este momento se deshecho la opción de utilizar un convertidor usb-paralelo establecer comunicación entre el portátil que carecía de puerto paralelo y la máquina de control numérico que sólo aceptaba comunicación paralela. Por ello tomamos la decisión de ejecutar la aplicación y realizar las pruebas en el ordenador del laboratorio que disponía de puerto paralelo.

Con respecto a las características del programa de comunicación con el puerto paralelo:

- Interfaz
  - En una primera versión disponía de 9 botones correspondientes a los pines de datos. Aunque sin funcionamiento
  - Un botón de enviar para enviar los datos por puerto paralelo. Aunque sin funcionamiento
  - Se estableció atajos de teclado en el que se enviaban pulsos para comunicarse con la máquina
- Implementación
  - Utilización de la librería inpout32.dll
  - Creación de la clase Paralelo y sus métodos
    - Inp32, Oup32
- Pruebas
  - Se realizaron numerosas pruebas de comunicación entre el programa y el puerto paralelo, cambiando configuraciones como la distancia entre flancos de subida y de bajada con variando los microsegundos de retardo.

El Programa de comunicación con el puerto paralelo tenia la siguiente interfaz:

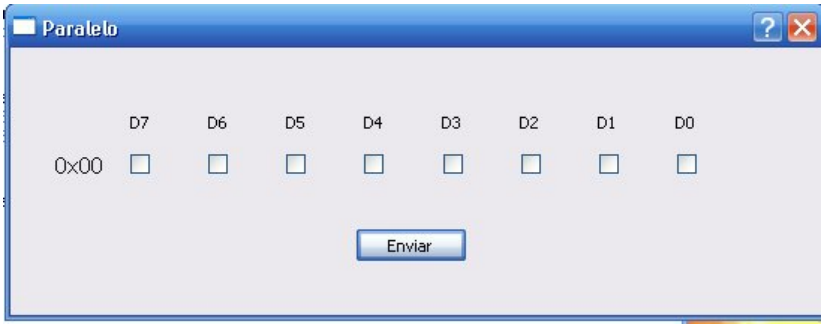

Figura 2.9: Paralelo v0

2.1.10. Incremento 10 : Puerto Paralelo hilos de ejecución

En este incremento se tomo la decisión de implementar el programa de comunicación con el puerto paralelo, mediante hilos de ejecución, para incrementar el tiempo de respuesta en el envió de los datos.

Los cambios que tuvieron lugar fueron los siguientes:

- Interfaz
  - Se eliminaron los botones de pines de datos.
  - Se eliminó el botón enviar.
  - Se crea botón de inicio de ejecución de la máquina.
  - Se crea botón de finalización de ejecución de la máquina.
- Implementación
  - Se crea la clase Pulsar que hereda la clase Qthread dedicada a la gestión de hilos de ejecución en memoria.
  - Se crean 2 métodos activar y desactivar, que envían por puerto paralelo los datos adecuado para llevar a cabo tal cometido con la máquina.
  - Se crea el método run, "cerebro" de la gestión de hilos de ejecución ya que se llama a esta función cuando se comienza a ejecutar un nuevo hilo.
  - Se incluye la clase Parar antes mencionada, para controlar los tiempos de retardo, aunque sin éxito.
- Pruebas
  - En este incremento se siguen realizando pruebas a diario con el cometido de poder establecer una comunicación correcta y fluida con la máquina de control numérico controlando el tiempo de cada pulso en microsegundos. En este incremento se logró la comunicación con la máquina de forma más fluida que en el anterior incremento.

El aspecto de esta segunda versión del programa de comunicación paralela era el siguiente:

Figura 2.10: Paralelo v1

2.1.11. Incremento 11 : Comunicación Paralela

En este incremento se cambió el sistema de comunicación entre la aplicación principal y la máquina de control numérico, instalando un sistema de intercambio de datos a través del puerto paralelo.

El 21 de diciembre de 2010 quede con mi tutor de proyecto para comentarle como iba el desarrollo del proyecto y su opinión al respecto.

En esta nueva versión se insertaron los siguientes cambios:

- Interfaz
  - Se introdujeron 4 nuevos botones de direccionamiento del movimiento de la máquina, para su desplazamiento en las 4 diagonales posibles.
  - Se elimina la comunicación serie, y se inserta la comunicación paralela
- Implementación
  - Se elimina la clase Comunicación
  - Se elimina la clase Pulsar
  - Se crea la clase Paralelo con hilos de ejecución
    - Creación de función Activar
    - Desactivar
    - run
  - MoverUpRight
  - MoverDownRignt
  - MoverUpLeft
  - MoverDownLeft
  - Se modifica función Conectar
  - Se modifica función Desconectar
  - Nuevos incrementos

- velX1()
- velX2()
- velX5()
- velX10()
- velX20()
- velX50()

- Se modifican las teclas de control de desplazamiento de la máquina

- Esta nueva versión se terminó de desarrollar el 26 de diciembre de 2010 y tenía el siguiente aspecto:

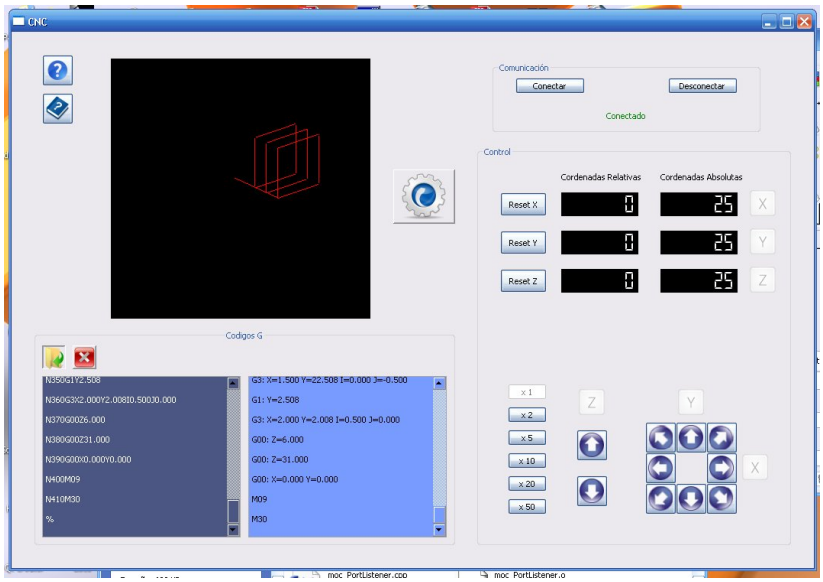

Figura 2.11: CNC v8

2.1.12. Incremento 12 : Documentación y manual

En este incremento se llevo a cabo principalmente un gran trabajo de documentación de todo lo realizado hasta el momento. Con respecto a la modificación de la aplicación , se realizó el manual de la aplicación, insertándolo en esta, para que se mostrara por medio del navegador web predeterminado del usuario. Se siguen realizando pruebas para perfeccionar el movimiento de la máquina cnc para ello se retocan las clases anteriormente implementadas.

Con respecto al interfaz no se llevaron a cabo ningún cambio reseñáble.

2.1.13. Incremento 13 : Control de pulsos y aplicación final

En este incremento, el principal cambio que se intrudujo fue, añadir el control de pulsos tanto de anchura de pulso como de velocidad. Otro de los cambios que se llevó a cabo en este incremento fue la de cambiar el limite de posición del eje x, y, z. También se introdujeron cambios en la interfaz del programa.Todos estos cambios dieron con la aplicación final, que se documentó debidamente en este incremento.

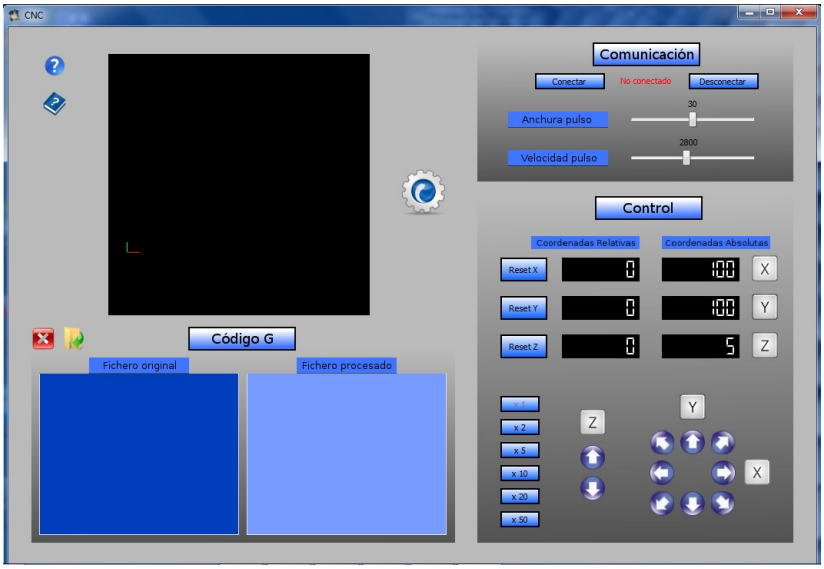

Figura 2.12: CNC v9

2.2. Diagrama de Gantt

Aquí mostramos el diagrama de Gantt para tener una percepción general de la duración del proyecto y sus incrementos realizados:

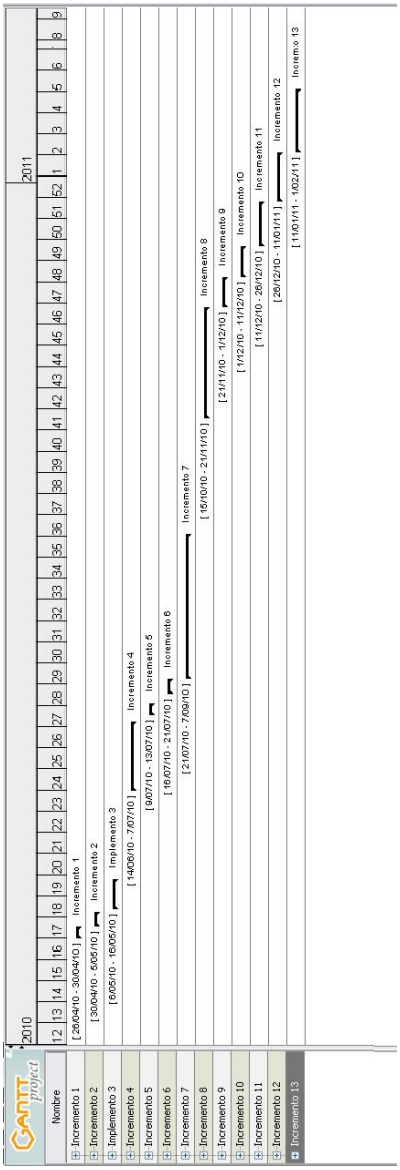

Figura 2.13: Diagrama Gantt General

Los diagramas más desglosados se especifican aquí abajo:

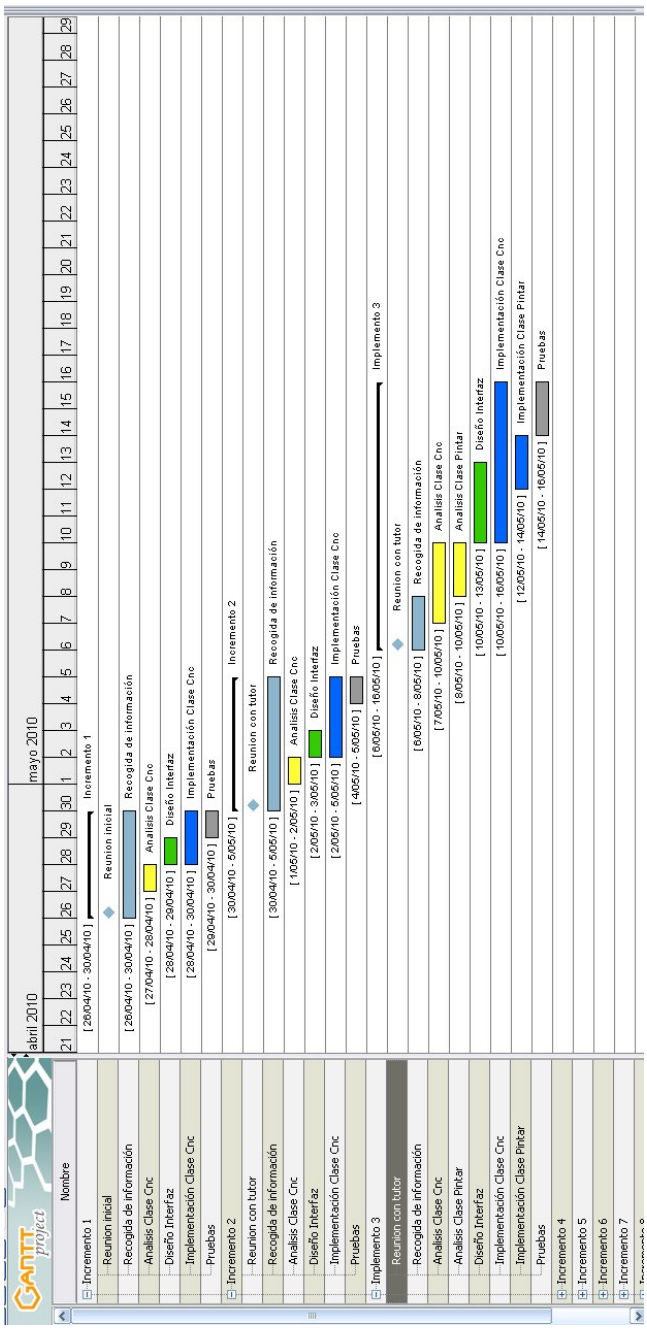

Figura 2.14: Diagrama Gantt I1 I2 I3

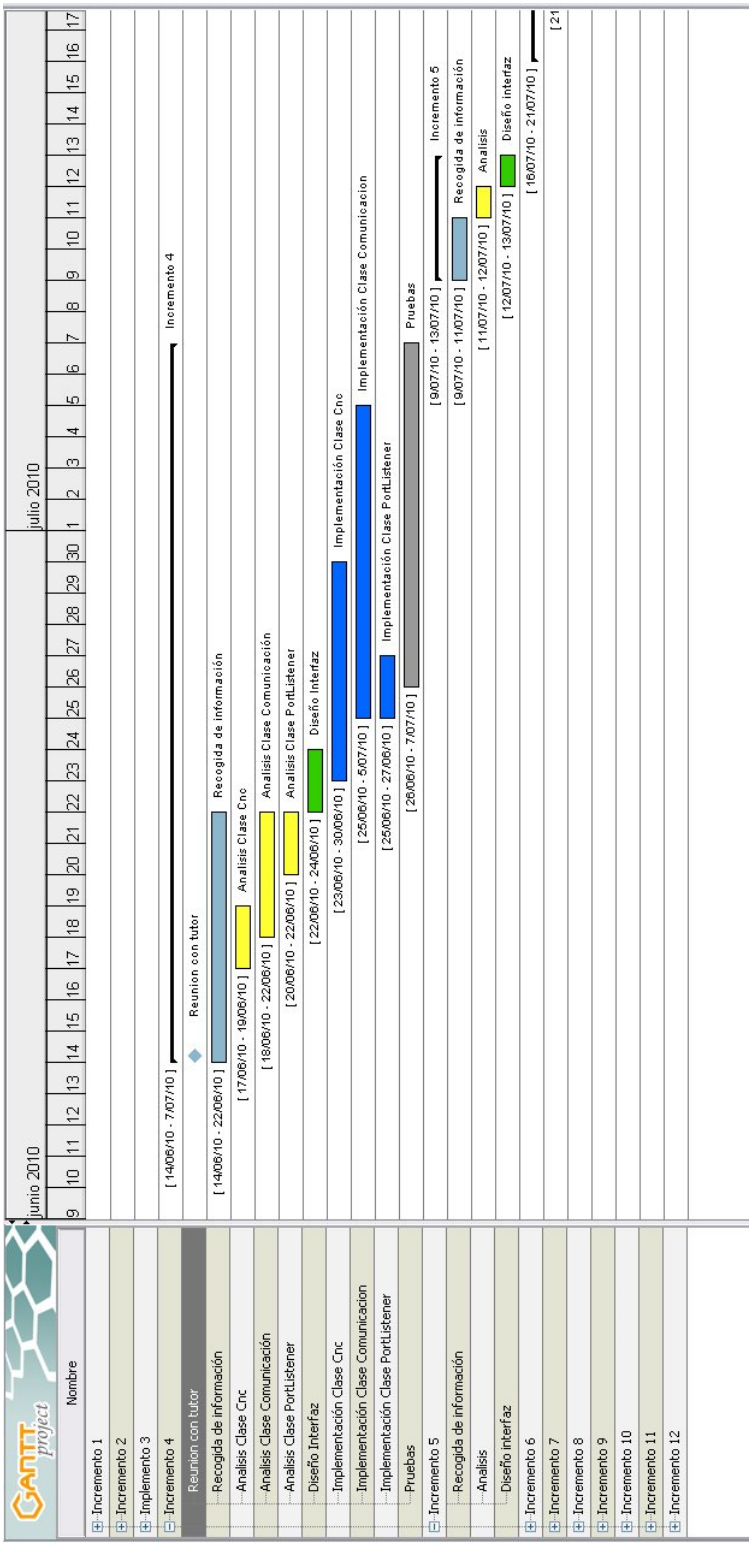

Figura 2.15: Diagrama Gantt I4 I5

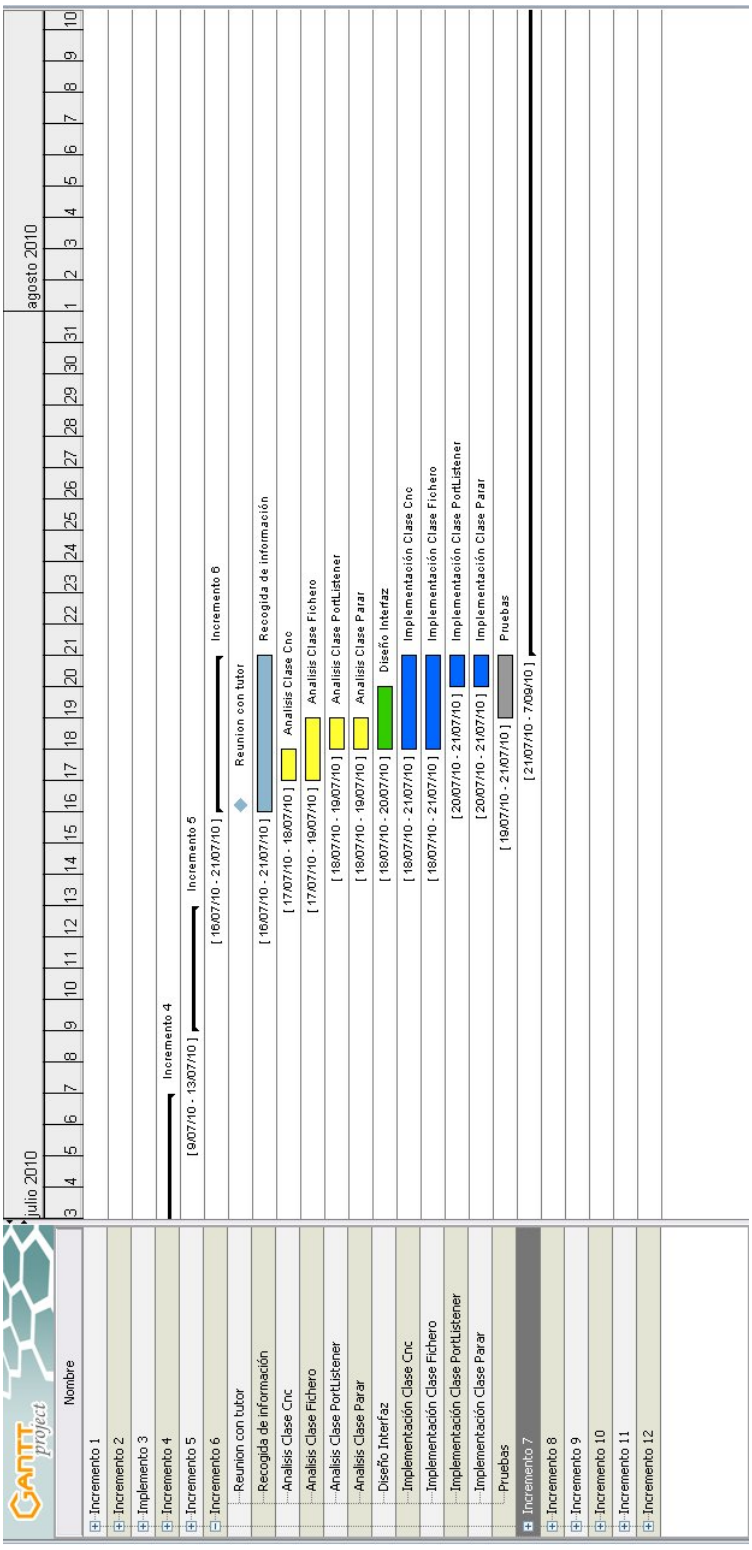

Figura 2.16: Diagrama Gantt I6

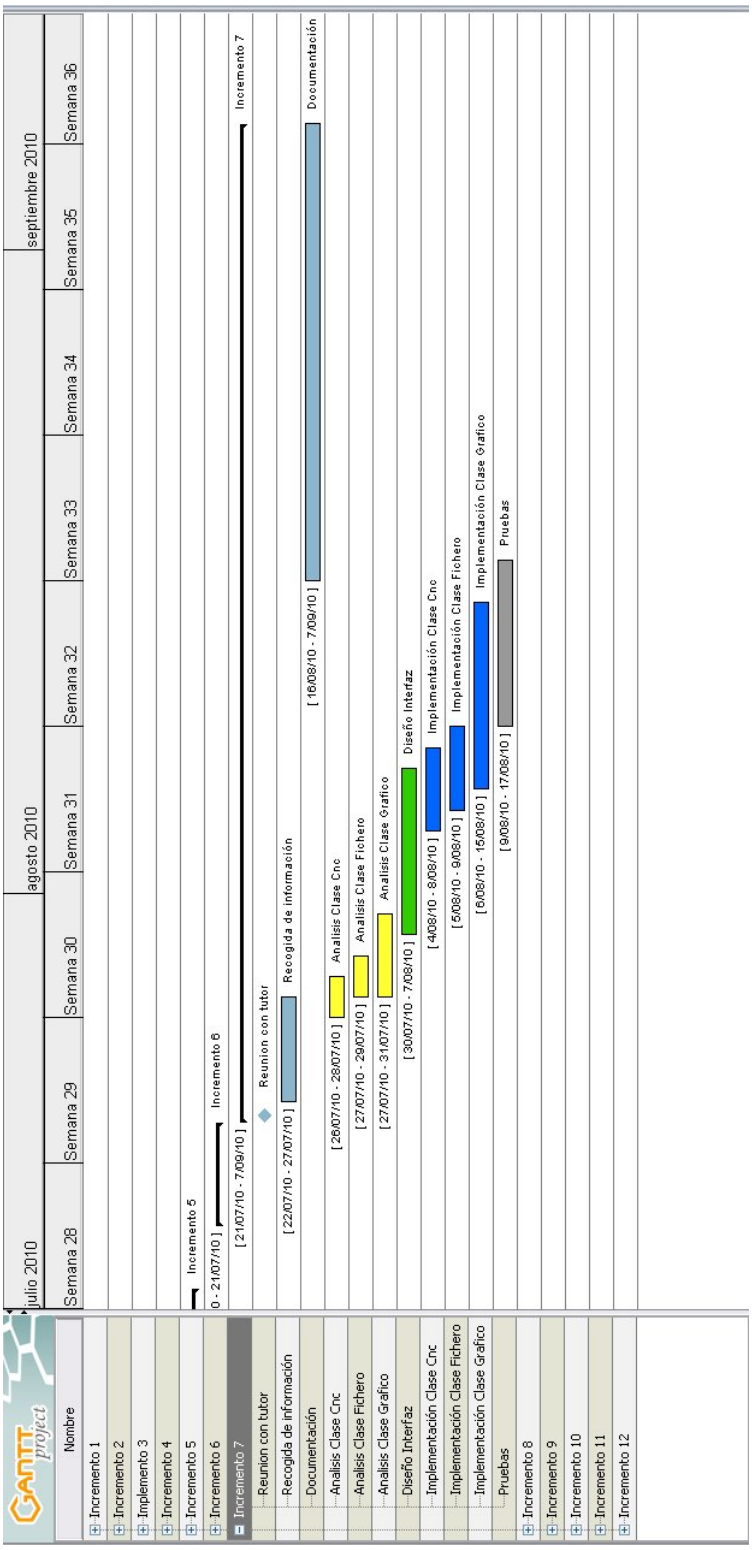

Figura 2.17: Diagrama Gantt I7

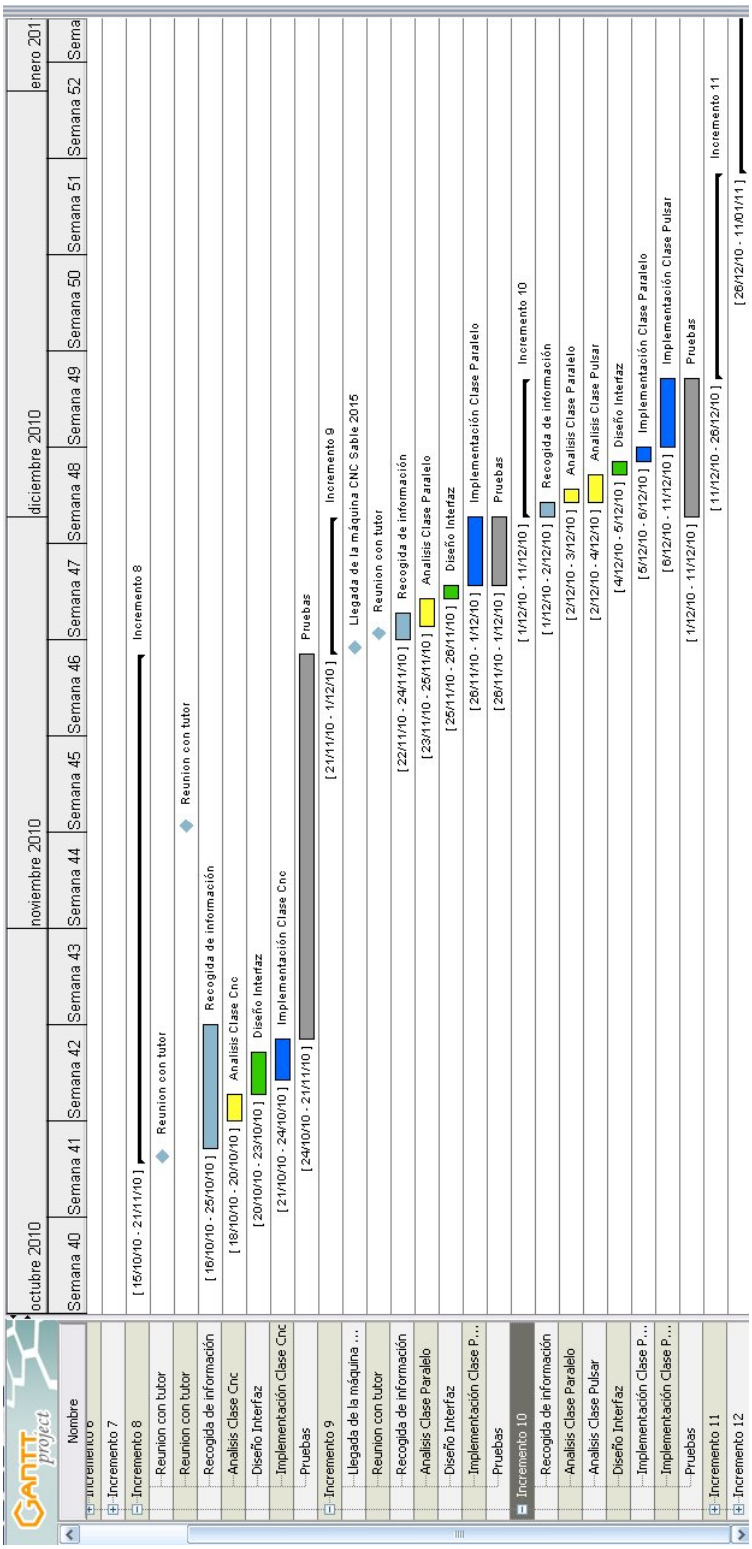

Figura 2.18: Diagrama Gantt I8 - I9 - I10

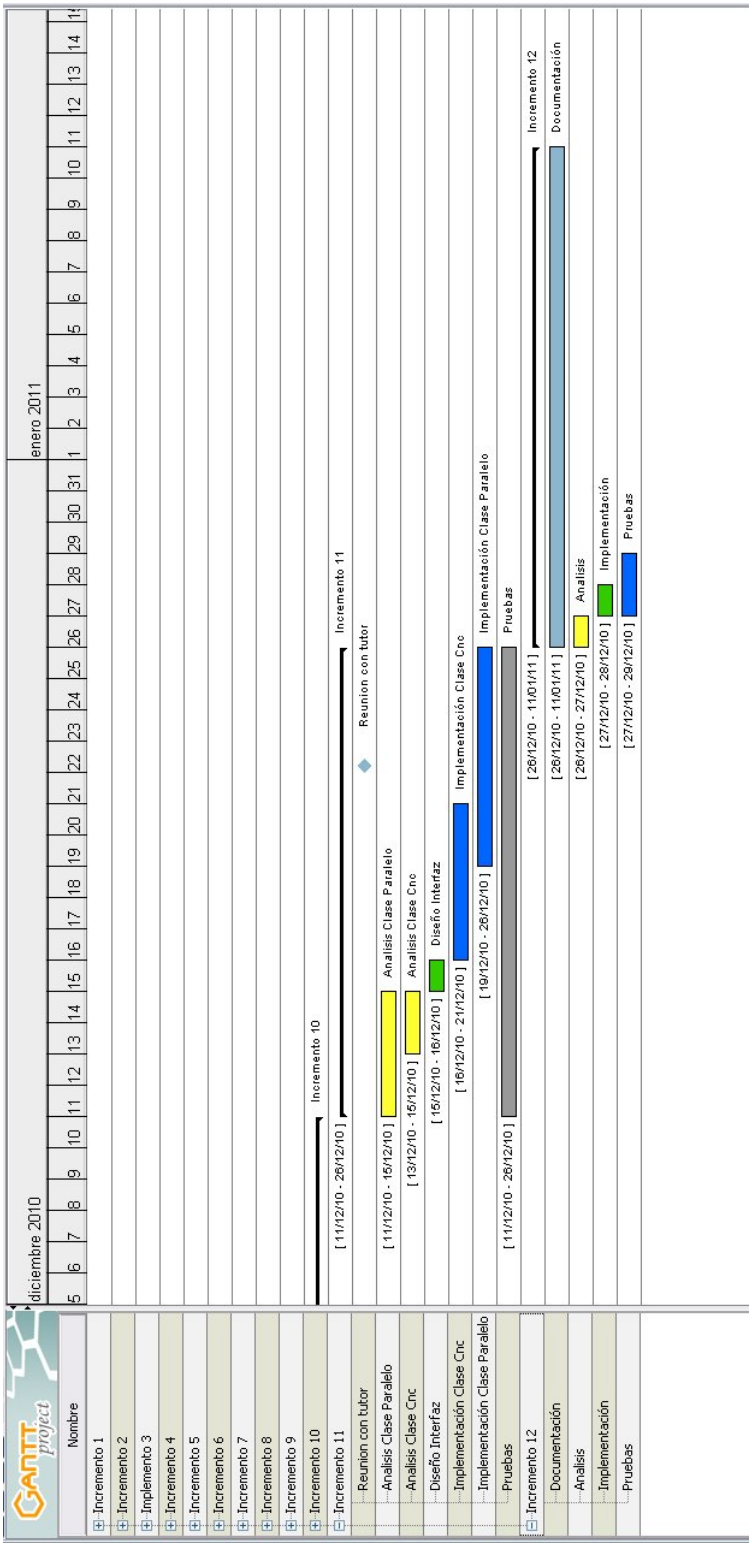

Figura 2.19: Diagrama Gantt I11 - I12

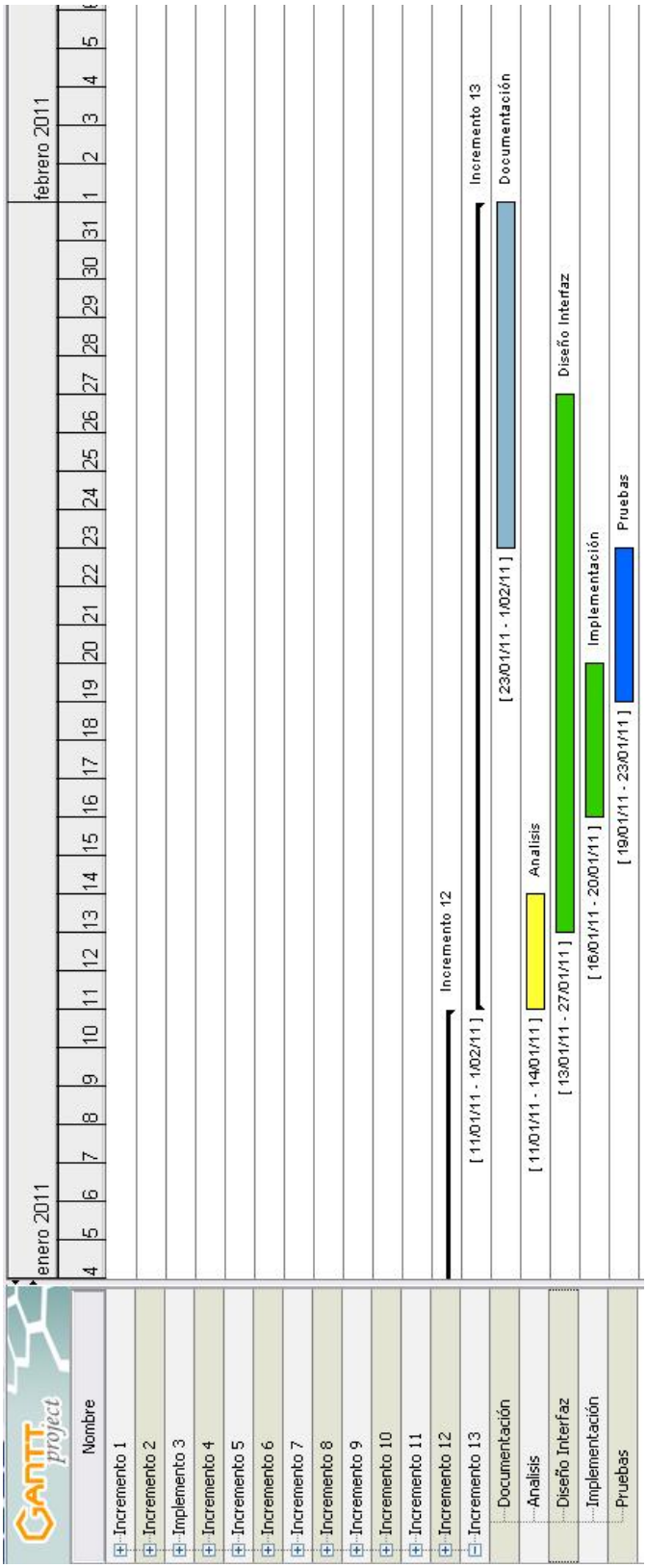

Figura 2.20: Diagrama Gantt I13

2.3. Esfuerzo y dedicación

En cuanto a los aspectos que más tiempo y esfuerzo se ha dedicado han sido a :

- Comunicación serie, todo lo relacionado con la comunicación serie llevo alrededor de 1 mes, en cuanto a la búsqueda de información, implementación y su posterior fase de pruebas.
- Comunicación paralela, todos los aspectos e comunicación paralela me llevaron casi 2 meses de pruebas con el analizador lógico, para entender y poder implementar todo lo relacionado con esto.
- Documentación, la labor de documentación me llevo bastante tiempo, ya que tuve que reunir la suficiente información para llevar a cabo toda la documentación del proyecto.

En conclusión podemos decir que las pruebas de comunicación se llevaron gran parte de tiempo del proyecto, tanto serie como paralela.

Capítulo 3

Especificación de requisitos

3.1. Introducción

3.1.1. Alcance

El nombre que se le ha dado a la aplicación final es: Desarrollo de una aplicación informática en tiempo real con entrada g-code para máquina de control numérico (CNC) en 3D.

Con esta aplicación se desea:

- Poder mover en tiempo real la máquina CNC en todas sus direcciones
- Poder representar gráficamente los ficheros con códigos Gs
- Poder procesar los ficheros proporcionados siempre y cuando tengan las siguientes características:
  - Cada instrucción comience por el comando Nxx siendo x el contador de instrucciones.
  - Los comandos dentro de cada instrucción no deben tener ningún carácter separador
  - Las medidas que utiliza son milímetros.
  - El fichero debe acabar con el carácter %
  - Los comandos aceptados por la aplicación son los siguientes.
    - G00
    - G01,G1
    - G03, G3 conteniendo este R o I y J además de X Y Z.
    - G90
    - G91
    - M03, M3

- M09, M9
  - Podemos ver una breve descripción de cada uno de ellos en el Apendice "Códigos G y Códigos M".

3.1.2. Definiciones, acrónimos y abreviaturas

- **CNC**: Control Numérico por Computador, también llamado CNC (en inglés Computer Numerical Control).
- **EMC**: Enhanced Machine Controller.
- **GNU**: GNU is Not Unix.
- **RAM**: Memoria de acceso aleatorio.
- **PC:** Computador personal.
- **EXE:** Abreviación del inglés executable, que se traduce en ejecutable.
- **UML:** Lenguaje Unificado de Modelado.
- **GUI:** La interfaz gráfica de usuario, del inglés graphical user interface.
- **QT:** biblioteca multiplataforma para desarrollar interfaces gráficas de usuario.

3.1.3. Estructura del documento

Este documentos de divide en 11 capítulos.

- **Introducción**: Este capitulo nos introduce en la temática del proyecto desarrollado.
- **Desarrollo del Calendario**: Este capitulo especifica las tareas realizadas así como el tiempo dedicado.
- **Análisis**: En este capitulo tratamos los aspectos relacionados con la fase de análisis.
- **Diseño**: En este capitulo tratamos los aspectos relacionados con la fase de diseño.
- **Implementación**: En este capitulo tratamos los aspectos relacionados con la fase de implementación.
- **Pruebas**: En este capitulo tratamos los aspectos relacionados con la fase de pruebas.
- **Conclusiones**: En este capitulo comentamos a la conclusiones llegadas tras la realización del proyecto.

- **Manual de Instalación**: En este capitulo documentamos como instalar el producto.
- **Manual de Usuario**: Se describe el manual de usuario.
- **Bibliografía y referencias**: En este capítulo describimos las fuentes en las que nos hemos basado para realizar el proyecto.
- **Códigos G y Códigos M:** En este anexo se enumeran los códigos estándar G y M
- **GNU Free Documentation License**: Licencia utilizada en la realización del proyecto.

3.2. Descripción general del proyecto

3.2.1. Perspectiva del producto

3.2.1.1. Interfaces del sistema

El sistema operativo sobre el que se debe implementar esta aplicación es sobre Windows, este sistema podría interactuar con otros sistemas dirigidos a la creación de ficheros cnc con instrucciones y comandos en códigos Gs.

3.2.1.2. Interfaces del usuario

La interfaz de usuario se basa en una única pantalla principal a partir de la cual se desarrolla toda la aplicación. La pantalla principal de la aplicación se compone de 4 partes bien diferenciadas y accesibles de manera intuitiva por parte del usuario que utilice la aplicación.

También se dispone de atajos de teclado para la interacción entre el usuario y la máquina de control numérico a través de la aplicación.

La aplicación consta de cuadros de dialogo para informar al usuario de cualquier problema o inconveniente que pudiera ocurrir mientras se está utilizando.

3.2.1.3. Requisitos de adaptación a la ubicación

Los requisitos de adaptación del producto en el sistema consisten en tener a disposición una máquina de control numérico de 3 ejes conectada a la corriente y a nuestro pc, para tener un uso correcto de la aplicación.

3.2.2. Funciones del producto

La función principal de la aplicación creada para el proyecto fin de carrera, es la de poder interactuar de manera fácil e intuitiva con una máquina de control numérico de 3 ejes.

Las características más importantes de esta aplicación son los siguientes:

- Capacidad de conectarse y vincularse a una máquina de control numérico.
- Posibilidad de mover los ejes de la máquina en tiempo real.
- Posibilidad de ver el desplazamiento efectuado por los ejes de la máquina tanto de manera absoluta como de manera relativa.
- La aplicación será capaz de abrir y filtrar un fichero con instrucciones en códigos G y M.
- La aplicación representa tales ficheros de forma gráfica y visual.

3.2.3. Características del usuario

El usuario que vaya a utilizar este producto deberá tener algunos conocimientos básicos tales como:

- Conocimiento básico de informática para utilizar su interfaz.
- Conocimiento de qué es una máquina de control numérico y de su funcionamiento básico.
- Conocimiento de códigos Gs estándar.
- Conocimiento de creación de ficheros cnc o conocimiento de aplicación externa como VCarve.

3.2.4. Restricciones

El proyecto se desarrolla utilizando el paradigma de la programación orientada a objetos.

El lenguaje de programación utilizado es libre, en este caso es C++ para todo el proyecto.

Se ha hecho uso de librerías gratuitas como son las librerías Qt.

Se ha utilizado una IDE libre y gratuita, como es la IDE Qt Creator.

El Sistema operativo que se debe utilizar es un sistema operativo Windows, ya que la comunicación mediante puerto paralelo para esta aplicación está desarrollado para tal sistema operativo.

Para la creación de este documento se ha utizado Latex, más concretamente se ha utilizado LyX.

3.2.5. Requisitos para futuras versiones del sistema

Para versiones futuras se intentará poder utilizar está aplicación bajo un sistema operativo linux.

Mejorar el movimiento de la máquina para que se produzca de manera más suave.

También sería interesante contar con alguna que otra más funcionalidad como por ejemplo poder editar el fichero cnc desde la aplicación, seleccionar la herramienta a utilizar por parte de la máquina o configurar la aceleración del movimiento de la máquina.

3.3. Requisitos específicos

3.3.1. Requisitos de interfaces externas

3.3.1.1. Requisitos de interfaces de usuario

Los requisitos de la interfaz gráfica del usuario son los siguientes:

- **Facilidad de comprensión, aprendizaje y uso:** En el caso de la intefaz gráfica del usuario utilizada para esta aplicación podemos decir que es de fácil comprensión, ya que sólo tenemos una única ventana principal con cada funcionalidad bien delimitada, por lo que consideramos es muy fácil de usar.
- **Diseño ergonómico mediante el iconos y botones:** En la aplicación podemos ver que tiene diferentes iconos que hacen más intuitiva si cabe la aplicación, además de la utilización de numerosos botones que ayudan a la comprensión de la aplicación.
- **Control desde teclado:** Se ha incluido en la aplicación la posibilidad de controlar la máquina además de la forma usual, es decir pulsando sobre los botones direccionales de la pantalla, con el uso del teclado, lo que hace más rápida y cómoda su ejecución.
- **Existencia de herramientas de Ayuda y Consulta:** La existencia de notas aclaratorias sobre botones significativos de la aplicación, se ha puesto a disposición del usuario el acceso al manual de la aplicación mediante un botón situado en la pantalla principal de la interfaz gráfica, lo que hace más accesible su posible consulta.
- **Tratamiento del error bien cuidado y adecuado al nivel de usuario**: La aplicación viene en el caso que el usuario quiera realizar cualquier operación no permitida, activará un cuadro de dialogo donde especificará qué paso ha de tomar para actuar de manera correcta.

3.3.2. Requisitos funcionales

Los requisitos funcionales del sistema agrupados por categorías son los siguientes:

3.3.2.1. Requisitos funcionales de comunicación

- El usuario puede conectarse y vincularse a la máquina de control numérico, siempre y cuando la máquina cnc esté conectada a la corriente y al pc donde se va a ejecutar la aplicación.
- El usuario puede desconectarse de la máquina de control numérico, estando esta previamente conectada y vinculada con la aplicación.

3.3.2.2. Requisitos funcionales de control

- El usuario podrá actuar sobre el motor del eje x moviéndolo en dirección positiva.
- El usuario podrá actuar sobre el motor del eje x moviéndolo en dirección negativa.
- El usuario podrá actuar sobre el motor del eje y moviéndolo en dirección positiva.
- El usuario podrá actuar sobre el motor del eje y moviéndolo en dirección negativa.
- El usuario podrá actuar sobre el motor del eje z moviéndolo en dirección positiva.
- El usuario podrá actuar sobre el motor del eje z moviéndolo en dirección negativa.
- El usuario podrá actuar conjuntamente sobre el motor del eje x y el motor del eje y moviendo ambos en dirección positiva.
- El usuario podrá actuar conjuntamente sobre el motor del eje x y el motor del eje y moviendo ambos en dirección negativa.
- El usuario podrá actuar conjuntamente sobre el motor del eje x en dirección positiva y el motor del eje y en dirección negativa.
- El usuario podrá actuar conjuntamente sobre el motor del eje y en dirección positiva y el motor del eje x en dirección negativa.

3.3.2.3. Requisitos funcionales de control de pulsos

- El usuario podrá aumentar el numero de pulsos dados al mover la máquina cnc, pudiendo tener 6 opciones diferentes, x1, x2, x5, x10, x20 y x50.
- El usuario podrá modificar la anchura de pulso enviada a la máquina de control numérico.
- El usuario podrá modificar la velocidad de pulso enviada a la máquina de control numérico.

3.3.2.4. Requisitos funcionales resetear posición relativa

- El usuario puede resetear la posición relativa del eje x, poniendo el desplazamiento de este en 0.00 mm.
- El usuario puede resetear la posición relativa del eje y, poniendo el desplazamiento de este en 0.00 mm.
- El usuario puede resetear la posición relativa del eje z, poniendo el desplazamiento de este en 0.00 mm.

3.3.2.5. Requisitos funcionales de códigos G

- El usuario podrá abrir un fichero cnc con instrucciones que contengan comandos Gs en el formato descrito anteriormente, procesándose y filtrándose en tal caso, y mostrándose en pantalla con el gráfico que representa tal fichero.
- El usuario podrá cerrar el fichero abierto con anterioridad.

3.3.2.6. Requisitos funcionales de ejecución

- El usuario podrá ejecutar el programa generado por el fichero cnc de entrada anteriormente abierto. Tal ejecución tendrá como resultado la ejecución de las instrucciones,entendibles por el programa por parte de la máquina de control numérico dando como resultado real la figura mostrada en el gráfico de la pantalla principal.

3.3.2.7. Requisitos funcionales de ayuda

- El usuario tendrá a su disposición un manual de ayuda que contendrá todas las opciones de la aplicación detalladas cuidadosamente para hacer más fácil la comprensión y utilización del programa por parte del usuario.
- El usuario podrá ver la información acerca de la aplicación.

3.3.3. Requisitos de rendimiento

El aspecto relacionado con el rendimiento en este caso es muy importante, porque de ello va a depender la ejecución en tiempo real de cualquier instrucción por parte de la máquina, para ello se ha creado hilos de ejecución en las funciones más críticas de la aplicación, como puede ser la transmisión de datos entre el ordenador y la máquina por el puerto paralelo.

3.3.4. Restricciones de diseño

En cuanto a las restricciones de diseño de esta aplicación, tenemos que decir que necesitamos que el fichero que recibe la aplicación para su posterior procesamiento y representación y ejecución, debe ser un fichero con ciertas características ya especificadas en el apartado alcance, en la especificación de requisitos del proyecto, si no recibimos el fichero en tal formato, el resultado puede no ser el esperado.

Otras de las restricciones de diseño de este proyecto es que la máquina de control numérico sea una máquina cnc de 3 ejes, y que este conectada al pc donde se esté ejecutando la aplicación, por medio del puerto paralelo LPT1.

3.3.5. Atributos de sistemas software

El sistema desarrollado para este proyecto debe seguir unas pautas imprescindibles tales como:

- **Fiabilidad:** El producto final debe ser fiable, ya que en un sistema con propósito de mover y automatizar una máquina de control numérico las operaciones ejecutadas por parte de la aplicación deben de ser lo más correctas posibles para que el resultado final sea totalmente fiable.
- **Disponibilidad:** El producto debe estar disponible en cualquier momento, es decir siempre que el usuario quiera utilizarlo, estará preparado para su uso y ejecución.
- **Mantenibilidad:** El producto debe ser fácil de mantener, ya que el sistema debe poder ser modificado para corregir fallos, o para mejorar su funcionamiento y otros atributos, además de esto, el producto debe poder adaptarse a cambios en el entorno, como puede ser en este caso el cambio de la máquina de control numérico.
- **Multilingüismo:** El producto es adaptable a las 2 lenguas más habladas en este planeta, tanto el español si es ejecutado en países hispanohablantes o en ingles si el programa es ejecutado en países angloparlantes.

Capítulo 4

Análisis

4.1. Metodología de desarrollo

La metodología del desarrollo utilizada en este proyecto se basa en MÉTRICA 3.

La metodología MÉTRICA Versión 3 proporciona un conjunto de métodos y técnicas que guía a los distintos profesionales de Sistemas y Tecnologías de la Información y Comunicaciones (STIC) en la obtención de los diversos productos de los procesos del ciclo de vida de un proyecto informático. Con el fin de mejorar la productividad de los distintos participantes y asegurar la calidad de los productos resultantes, la mayoría de las técnicas propuestas están soportadas por herramientas disponibles en el mercado que automatizan en mayor o menor grado su utilización. En cualquier caso, no todos los productos resultantes de cada tarea son susceptibles de obtenerse de forma automatizada.

La metodología del desarrollo utilizada para la elaboración de este proyecto se basa en el Proceso Unicado Racional (Rational Unied Process (RUP)), este proceso de desarrollo de software junto con el Lenguaje Unicado de Modelado (UML) constituyen la metodología estándar mas utilizada para el análisis, implementación y documentación de sistemas orientados a objetos.

Rational Unied Process (RUP) se utiliza para el desarrollo de sistemas orientados a objetos. Esta basado en el Unied Process (UP) y utiliza Unied Modeling Languaje (UML) como notación. Este proyecto utiliza la metodología orientada a objetos, por lo tanto se aplicara Rational Unied Process para la metodología del desarrollo.

El modelo de ciclo de vida utilizado para la elaboración de este proyecto ha sido el modelo incremental.

El modelo incremental tiene las siguientes características:

El modelo incremental combina elementos del modelo lineal secuencial con la filosofía interactiva del modelo de construcción de prototipos. En el modelo incremental se crean sucesivas versiones del software hasta obtener el producto final. Cada versión del software se obtiene añadiendo nuevas funciones o requisitos (incrementos) a la versión obtenida en la iteración anterior. El software no se ve como un producto que se entrega en una determinada fecha, sino como la integración de las diferentes versiones del producto obtenidas en cada iteración.

El modelo de proceso incremental, como el modelo de construcción de prototipos y otros enfoques evolutivos, es interactivo. Pero a diferencia de la construcción de prototipos, el modelo incremental se basa en la entrega de un producto operacional en cada incremento. Los primeros incrementos son versiones reducidas del producto final y proporcionan al cliente una base para la evaluación.

En este proyecto se ha decidido utilizarle modelo incremental por las siguientes razones:

- Al comienzo del proyecto existía una alta incertidumbre en los requisitos del sistema, ya que no conociamos con exactitud cuales eran todas las funcionalidades que iba a tener nuestro sistema.
- En el desarrollo del producto existieron riesgos técnicos, como fue la necesidad de un nuevo hardware del que no disponíamos, mas concretamente, la máquina de control numérico, además de ello, no sabíamos en que fecha íbamos a tenerla disponible.

4.2. Modelo de casos de uso

4.2.1. Diagrama de casos de uso general

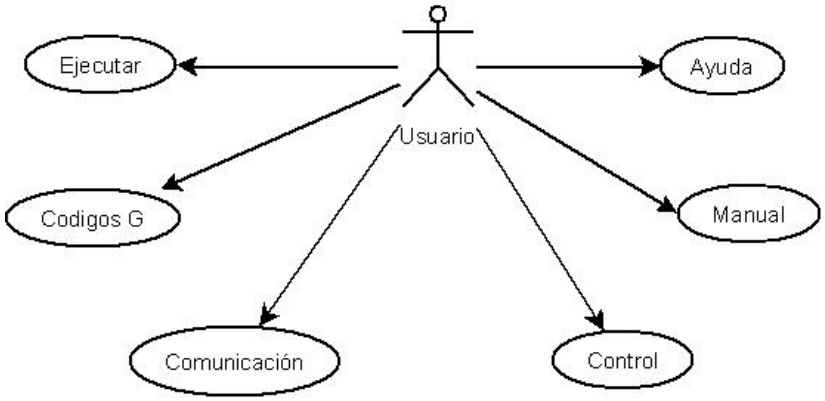

Figura 4.1: Diagrama Casos de Uso General

4.2.1.1. Especificación caso de uso Ejecutar

Identificación de escenarios

- **Escenario Principal:** Ejecutar OK.
- **Escenario Error:** El fichero no está abierto.
- **Escenario Error2:** El sistema no está conectado.

Descripción de caso de uso

- **Descripción:** Se ejecuta el fichero cargado
- **Actores: Usuario**
- **Flujos Principales**

1. El usuario quiere ejecutar el fichero.
2. El sistema comprueba que el fichero está abierto.
3. El sistema crea un vector con los códigos anteriormente filtrados.
4. El sistema crea 3 variables con la posición actual de la herramienta de la máquina.
5. El sistema comprueba que existe conexión con la máquina.
6. El sistema establece el movimiento mínimo de la máquina en 1 mm
7. El sistema comienza a recorrer el vector

8. El sistema comprueba que la linea contiene el comando G00
9. El sistema establece el movimiento mínimo de la máquina en 1 mm
10. El sistema crea 3 variables con las posiciones destino.
11. El sistema crea 3 variables con las distancias entre los puntos de origen y destino.
12. El sistema crea 3 variables para el incremento de la posición de x, y, z.
13. El sistema comprueba el valor de x no es nulo ('$').
14. El sistema comprueba que no estamos en posiciones absolutas.
15. El sistema asigna a la posición destino la suma de la posición inicial + el valor x de la linea.
16. El sistema comprueba el valor de y no es nulo ('$').
17. El sistema comprueba que no estamos en posiciones absolutas.
18. El sistema asigna a la posición destino la suma de la posición inicial + el valor y de la linea.
19. El sistema comprueba el valor de z no es nulo ('$').
20. El sistema comprueba que no estamos en posiciones absolutas.
21. El sistema asigna a la posición destino la suma de la posición inicial + el valor z de la linea.
22. El sistema actualiza el valor de la distancias entre origen y destino de 'x', 'y' y 'z'
23. El sistema calcula la distancia euclídea a recorrer por la máquina en el espacio XY
24. El sistema establece los pasos intermedios que tiene tiene que realizar la máquina, dividiendo la distancia de cada eje entre la distancia euclídea
25. El sistema establece los pulsos enviados en cada iteración correspondientes a la distancia a recorrer
26. El sistema comprueba que la distancia a recorrer en el eje z es mayor a 0.
27. El sistema establece los pulsos a enviar para el movimiento del eje z.
28. El sistema comprueba que el destino es mayor al origen y establece la dirección a tomar.
29. El sistema comienza la ejecución del hilo dedicado al movimiento de la máquina.
30. El sistema comprueba si la dirección a tomar, es en x estricto, y estricto o en diagonal de 45^{o}
31. El sistema comprueba que la máquina tiene que moverse en diagonal

32. El sistema establece los pasos a enviar a la máquina
33. El sistema comprueba que diagonal tomar dependiendo de la posición de destino x y
34. El sistema comienza la ejecución del hilo dedicado al movimiento de la máquina
35. El sistema actualiza los valores de posiciones relativas y absolutas
36. El sistema muestra el nuevo emplazamiento absoluto
37. El sistema muestra el nuevo emplazamiento relativo

- **Flujos Alternativos**

2a El fichero no está abierto

1. El sistema emite el error mediante un cuadro de dialogo

5a El sistema no está conectado

1. El sistema emite el error mediante un cuadro de dialogo

13a El sistema asigna a la posición destino el valor actual de x

14a El sistema asigna a la posición destino el valor x de la linea

16a El sistema asigna a la posición destino el valor actual de y

15a El sistema asigna a la posición destino el valor y de la linea

19a El sistema asigna a la posición destino el valor actual de z

20a El sistema asigna a la posición destino el valor z de la linea

30a El sistema comprueba que la direccion a tomar no es x estricto,o no es y estricto o no es en diagonal de 45º

1. El sistema comprueba que distancia a recorrer en x es mayor que 0
2. El sistema establece los pasos a enviar a la máquina
3. El sistema comprueba que dirección tomar dependiendo de la posicion de destino x
4. El sistema comienza la ejecucion del hilo dedicado al movimiento de la máquina
5. El sistema comprueba que la distancia a recorrer en y es mayor a 0
6. El sistema establece los pasos a enviar a la máquina
7. El sistema comprueba que dirección tomar dependiendo de la posicion de destino y
8. El sistema comienza la ejecucion del hilo dedicado al movimiento de la máquina
9. Repetir pasos 1-8 mientras quedan pasos por realizar

31a El sistema comprueba que la distancia x es mayor a 0

1. El sistema establece los pasos a enviar a la máquina
2. El sistema comprueba que dirección tomar dependiendo de la posicion de destino x
3. El sistema comienza la ejecucion del hilo dedicado al movimiento de la máquina
4. El sistema comprueba que la distancia x es mayor a 0
5. El sistema establece los pasos a enviar a la máquina
6. El sistema comprueba que dirección tomar dependiendo de la posicion de destino y
7. El sistema comienza la ejecucion del hilo dedicado al movimiento de la máquina

8a El sistema comprueba que la linea contiene el comando G1.

1. El sistema establece el movimiento minimo de la máquina en 1 mm
2. El sistema crea 3 variables con las posiciones destino.
3. El sistema crea 3 variables con las distancias entre los puntos de origen y destino.
4. El sistema crea 3 variables para el incremento de la posición de x, y, z.
5. El sistema comprueba el valor de x no es nulo ('$').
6. El sistema comprueba que no estamos en posiciones absolutas.
7. El sistema asigna a la posicion destino la suma de la posicion inicial + el valor x de la linea.
8. El sistema comprueba el valor de y no es nulo ('$').
9. El sistema comprueba que no estamos en posiciones absolutas.
10. El sistema asigna a la posicion destino la suma de la posicion inicial + el valor y de la linea.
11. El sistema comprueba el valor de z no es nulo ('$').
12. El sistema comprueba que no estamos en posiciones absolutas.
13. El sistema asigna a la posicion destino la suma de la posicion inicial + el valor z de la linea.
14. El sistema actualiza el valor de la distancias entre origen y destino de 'x', 'y' y 'z'
15. El sistema calcula la distancia euclidea a recorrer por la máquina en el espacio XY
16. El sistema establece los pasos intermedios que tiene tiene que realizar la máquina, dividiendo la distancia de cada eje entre la distancia euclidea
17. El sistema establece los pulsos enviados en cada iteración correspondientes a la distancia a recorrer
18. El sistema comprueba que la distancia a recorrer en el eje z es mayor a 0.

19. El sistema establece los pulsos a enviar para el movimiento del eje z.
20. El sistema comprueba que el destino es mayor al origen y establece la dirección a tomar.
21. El sistema comienza la ejecución del hilo dedicado al movimiento de la máquina.
22. El sistema comprueba si la direccion a tomar, es en x estricto, y estricto o en diagonal de 45^{o}
23. El sistema comprueba que la máquina tiene que moverse en diagonal
24. El sistema establece los pasos a enviar a la máquina
25. El sistema comprueba que diagonal tomar dependiendo de la posicion de destino x y
26. El sistema comienza la ejecucion del hilo dedicado al movimiento de la máquina

22a El sistema comprueba que la direccion a tomar no es x estricto,o no es y estricto o no es en diagonal de 45^{o}

- *a*) El sistema comprueba que distancia a recorrer en x es mayor que 0
- *b*) El sistema establece los pasos a enviar a la máquina
- *c*) El sistema comprueba que dirección tomar dependiendo de la posicion de destino x
- *d*) El sistema comienza la ejecucion del hilo dedicado al movimiento de la máquina
- *e*) El sistema comprueba que la distancia a recorrer en y es mayor a 0
- *f*) El sistema establece los pasos a enviar a la máquina
- *g*) El sistema comprueba que dirección tomar dependiendo de la posicion de destino y
- *h*) El sistema comienza la ejecucion del hilo dedicado al movimiento de la máquina
- *i*) Repetir pasos 1-8 mientras quedan pasos por realizar

23a El sistema comprueba que la distancia x es mayor a 0

- *a*) El sistema establece los pasos a enviar a la máquina
- *b*) El sistema comprueba que dirección tomar dependiendo de la posicion de destino x
- *c*) El sistema comienza la ejecucion del hilo dedicado al movimiento de la máquina
- *d*) El sistema comprueba que la distancia x es mayor a 0
- *e*) El sistema establece los pasos a enviar a la máquina
- *f*) El sistema comprueba que dirección tomar dependiendo de la posicion de destino y

g) El sistema comienza la ejecucion del hilo dedicado al movimiento de la máquina

27. El sistema actualiza los valores de posiciones relativas y absolutas
28. El sistema muestra el nuevo emplazamiento absoluto
29. El sistema muestra el nuevo emplazamiento relativo

8b El sistema comprueba que la linea contiene el comando G3.

1. El sistema establece el movimiento minimo de la máquina en 1 mm
2. El sistema crea 3 variables con las posiciones destino.
3. El sistema crea 3 variables con las distancias entre los puntos de origen y destino.
4. El sistema crea 3 variables para el incremento de la posición de x, y, z.
5. El sistema comprueba el valor de x no es nulo ('$').
6. El sistema comprueba que no estamos en posiciones absolutas.
7. El sistema asigna a la posicion destino la suma de la posicion inicial + el valor x de la linea.
8. El sistema comprueba el valor de y no es nulo ('$').
9. El sistema comprueba que no estamos en posiciones absolutas.
10. El sistema asigna a la posicion destino la suma de la posicion inicial + el valor y de la linea.
11. El sistema comprueba el valor de z no es nulo ('$').
12. El sistema comprueba que no estamos en posiciones absolutas.
13. El sistema asigna a la posicion destino la suma de la posicion inicial + el valor z de la linea.
14. El sistema comprueba que el valor de r no es nulo ('$').
15. El sistema asigna el valor del radio como suma de la posición inicial + el valor r de la linea
16. El sistema comprueba que el valor de ri no es nulo ('$').
17. El sistema asigna el valor del radio i como suma de la posición inicial + el valor ri de la linea
18. El sistema comprueba que el valor de rj no es nulo ('$').
19. El sistema asigna el valor del radio como suma de la posición inicial + el valor rj de la linea
20. El sistema comprueba que r es nulo ('$') y lo calcula.
21. El sistema calcula los angulos alfa y beta que forman el arco
22. El sistema comprueba que el arco está en el 3er o 4o cuadrante y le sumamos 180
23. El sistema le suma 360^{o} a alfa si es menor que beta
24. El sistema establece los valores de i y j
25. El sistema establece las posiciones absolutas x e y

26. El sistema establece los valores de i y j
27. El sistema establece los valores de x e y
28. El sistema calcula la posición de destino x e y
29. El sistema establece las nuevas distancias x e y
30. El sistema calcula la distancia euclidea a recorrer por la máquina
31. El sistema establece los pasos intermedios que tiene tiene que realizar la máquina, dividiendo la distancia de cada eje entre la distancia euclidea si esta no es 0.
32. El sistema establece la distancia a recorrer en cada iteración
33. El sistema comprueba si la direccion a tomar, es en x estricto, y estricto o en diagonal de 45º
34. El sistema comprueba que la máquina tiene que moverse en diagonal
35. El sistema establece los pasos a enviar a la máquina
36. El sistema comprueba que diagonal tomar dependiendo de la posicion de destino x y
37. El sistema comienza la ejecucion del hilo dedicado al movimiento de la máquina

33a El sistema comprueba que la direccion a tomar no es x estricto,o no es y estricto o no es en diagonal de 45º

a) El sistema comprueba que distancia a recorrer en x es mayor que 0
b) El sistema establece los pasos a enviar a la máquina
c) El sistema comprueba que dirección tomar dependiendo de la posicion de destino x
d) El sistema comienza la ejecucion del hilo dedicado al movimiento de la máquina
e) El sistema comprueba que la distancia a recorrer en y es mayor a 0
f) El sistema establece los pasos a enviar a la máquina
g) El sistema comprueba que dirección tomar dependiendo de la posicion de destino y
h) El sistema comienza la ejecucion del hilo dedicado al movimiento de la máquina
i) Repetir pasos 1-8 mientras quedan pasos por realizar

34a El sistema comprueba que la distancia x es mayor a 0

a) El sistema establece los pasos a enviar a la máquina
b) El sistema comprueba que dirección tomar dependiendo de la posicion de destino x
c) El sistema comienza la ejecucion del hilo dedicado al movimiento de la máquina
d) El sistema comprueba que la distancia x es mayor a 0

e) El sistema establece los pasos a enviar a la máquina

f) El sistema comprueba que dirección tomar dependiendo de la posicion de destino y

g) El sistema comienza la ejecucion del hilo dedicado al movimiento de la máquina

38. El sistema actualiza los valores de posiciones x,y de origen y x,y de destino
39. El sistema repite los pasos 28-37 mientras haya arco que recorrer
40. El sistema actualiza los valores de posiciones relativas y absolutas
41. El sistema muestra el nuevo emplazamiento absoluto
42. El sistema muestra el nuevo emplazamiento relativo

4.2.1.2. Especificación caso de uso Ayuda

Identificación de escenarios

- **Escenario Principal:** Ayuda OK

Descripción de caso de uso

- **Descripción:** Se muestra la ayuda de la aplicación
- **Actores:** Usuario
- **Flujos Principales**

1. El sistema crea un cuadro de dialogo con la información de la aplicación y la muestra.

4.2.1.3. Especificación caso de uso Manual

Identificación de escenarios

- **Escenario Principal:** Manual OK

Descripción de caso de uso

- **Descripción:** Se muestra el manual de la aplicación
- **Actores:** Usuario
- **Flujos Principales**

1. El sistema abre el enlace que lleva al manual de la aplicación, mostrando este en el navegador predeterminado del usuario.

4.2.2. Diagrama de casos de uso : Comunicación

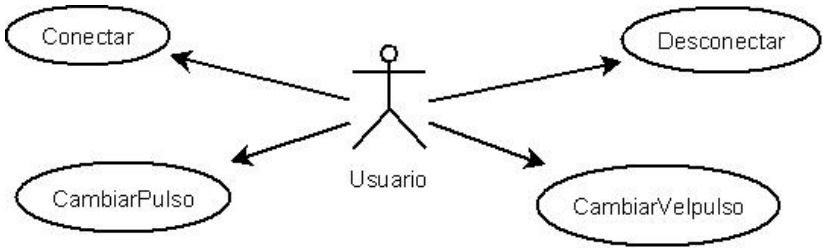

Figura 4.2: Diagrama de caso de uso : Comunicación

4.2.2.1. Especificación caso de uso Conectar

Identificación de escenarios

- **Escenario Principal:** Conectar OK

Descripción de caso de uso

- **Descripción:** Conectamos y activamos con la máquina
- **Actores:** Usuario
- **Reconducción:** Ninguna
- **Post condición:** Se establece la conexión con la máquina CNC con éxito
- **Flujos Principales**

  1. El usuario quiere conectarse con la máquina CNC
  2. El sistema comprueba que anchura y velocidad de pulso esta marcada
  3. El sistema establece la cantidad de pulsos en cada movimiento dependiendo lo que esté establecido por el usuario
  4. El sistema activa la máquina
  5. El sistema establece que está conectado y lo muestra

4.2.2.2. Especificación caso de uso Desconectar

Identificación de escenarios

- **Escenario Principal:** Desconectar OK
- **Escenario Error:** El sistema no está conectado

Descripción de caso de uso

- **Descripción:** Desconectamos el sistema de la máquina

- **Actores:** Usuario
- **Precondición:** El sistema está conectado
- **Postcondición:** Desconexión del sistema con la máquina CNC realizado con éxito
- **Flujos Principales**

  1. El usuario quiere desconectar la máquina del sistema
  2. El sistema comprueba que está conectado actualmente a la máquina
  3. El sistema desactiva la máquina CNC
  4. El sistema establece el sistema como desconectado
  5. El sistema muestra la desconexión

- **Flujos Alternativos**

  2a El sistema no está conectado
  1. El sistema emite error

4.2.2.3. Especificación caso de uso CambiarPulso

Identificación de escenarios

- **Escenario Principal:** CambiarPulso OK

Descripción de caso de uso

- **Descripción:** Cambiamos la anchura de pulsos enviados a la máquina
- **Actores:** Usuario
- **Reconducción:** Ninguna
- **Post condición:** Se establece la anchura de pulsos enviados a la máquina CNC de forma exitosa
- **Flujos Principales**

  1. El usuario quiere cambiar la anchura de pulsos.
  2. El sistema comprueba cual es la anchura de pulsos elegida por el usuario, la establece y la muestra.

4.2.2.4. Especificación caso de uso CambiarVelpulso

Identificación de escenarios

- **Escenario Principal:** CambiarVelpulso OK

Descripción de caso de uso

- **Descripción:** Cambiamos la velocidad de pulsos enviados a la máquina
- **Actores:** Usuario
- **Reconducción:** Ninguna
- **Post condición:** Se establece la velocidad de pulsos enviados a la máquina CNC de forma exitosa
- **Flujos Principales**

  1. El usuario quiere cambiar la velocidad de pulsos.
  2. El sistema comprueba cual es la velocidad de pulsos elegida por el usuario, la establece y la muestra.

4.2.3. Diagrama de casos de uso : Control

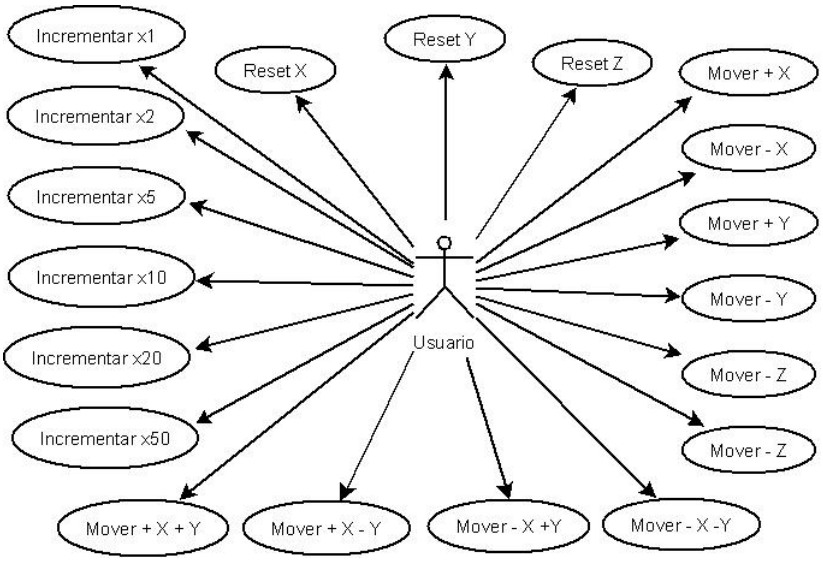

Figura 4.3: Diagrama de caso de uso : Control

4.2.3.1. Especificación caso de uso Reset X

Identificación de escenarios

- **Escenario Principal:** Reset X OK

Descripción de caso de uso

- **Descripción:** Reseteamos la posición relativas X
- **Actores:** Usuario
- **Precondición:** Ninguna
- **Postcondición:** Se pone a 0 el valor de la posición relativa del eje X
- **Flujos Principales**
  1. El usuario quiere resetear X
  2. El sistema pone a 0 el valor de la posición relativa actual del eje X

4.2.3.2. Especificación caso de uso Reset Y

Identificación de escenarios

- **Escenario Principal:** Reset Y OK

Descripción de caso de uso

- **Descripción:** Reseteamos la posición relativas Y
- **Actores:** Usuario
- **Precondición:** Ninguna
- **Postcondición:** Se pone a 0 el valor de la posición relativa del eje Y
- **Flujos Principales**
  1. El usuario quiere resetear Y
  2. El sistema pone a 0 el valor de la posición relativa actual del eje Y

4.2.3.3. Especificación caso de uso Reset Z

Identificación de escenarios

- **Escenario Principal:** Reset Z OK

Descripción de caso de uso

- **Descripción:** Reseteamos la posición relativas Z
- **Actores:** Usuario
- **Precondición:** Ninguna
- **Postcondición:** Se pone a 0 el valor de la posición relativa del eje Z
- **Flujos Principales**
  1. El usuario quiere resetear Z
  2. El sistema pone a 0 el valor de la posición relativa actual del eje Z

4.2.3.4. Especificación caso de uso Incrementar x1

Identificación de escenarios

- **Escenario Principal:** Incrementar x1 OK

Descripción de caso de uso

- **Descripción:** Establecemos el incremento x1
- **Actores:** Usuario
- **Precondición:** Ninguna
- **Postcondición:** Se establece el incremento en 0.1 mm
- **Flujos Principales**

  1. El usuario quiere establecer el incremento en x1
  2. El sistema establece el incremento en 0.1
  3. El sistema deshabilita la opción x1 y habilita las demás

4.2.3.5. Especificación caso de uso Incrementar x2

Identificación de escenarios

- **Escenario Principal:** Incrementar x2 OK

Descripción de caso de uso

- **Descripción:** Establecemos el incremento x2
- **Actores:** Usuario
- **Precondición:** Ninguna
- **Postcondición:** Se establece el incremento en 0.2 mm
- **Flujos Principales**

  1. El usuario quiere establecer el incremento en x2
  2. El sistema establece el incremento en 0.2
  3. El sistema deshabilita la opción x2 y habilita las demás

4.2.3.6. Especificación caso de uso Incrementar x5

Identificación de escenarios

- **Escenario Principal:** Incrementar x5 OK

Descripción de caso de uso

- **Descripción:** Establecemos el incremento x5
- **Actores:** Usuario
- **Precondición:** Ninguna
- **Postcondición:** Se establece el incremento en 0.5 mm
- **Flujos Principales**

  1. El usuario quiere establecer el incremento en x5
  2. El sistema establece el incremento en 0.5
  3. El sistema deshabilita la opción x5 y habilita las demás

4.2.3.7. Especificación caso de uso Incrementar x10

Identificación de escenarios

- **Escenario Principal:** Incrementar x10 OK

Descripción de caso de uso

- **Descripción:** Establecemos el incremento x10
- **Actores:** Usuario
- **Precondición:** Ninguna
- **Postcondición:** Se establece el incremento en 1 mm
- **Flujos Principales**

  1. El usuario quiere establecer el incremento en x10
  2. El sistema establece el incremento en 1
  3. El sistema deshabilita la opción x10 y habilita las demás

4.2.3.8. Especificación caso de uso Incrementar x20

Identificación de escenarios

- **Escenario Principal:** Incrementar x20 OK

Descripción de caso de uso

- **Descripción:** Establecemos el incremento x20
- **Actores:** Usuario
- **Precondición:** Ninguna
- **Postcondición:** Se establece el incremento en 2.00 mm
- **Flujos Principales**

  1. El usuario quiere establecer el incremento en x20
  2. El sistema establece el incremento en 2.00
  3. El sistema deshabilita la opción x20 y habilita las demás

4.2.3.9. Especificación caso de uso Incrementar x50

Identificación de escenarios

- **Escenario Principal:** Incrementar x50 OK

Descripción de caso de uso

- **Descripción:** Establecemos el incremento x50
- **Actores:** Usuario
- **Precondición:** Ninguna
- **Postcondición:** Se establece el incremento en 5.00 mm
- **Flujos Principales**

  1. El usuario quiere establecer el incremento en x50
  2. El sistema establece el incremento en 5.00
  3. El sistema deshabilita la opción x50 y habilita las demás

4.2.3.10. Especificación caso de uso Mover + X

Identificación de escenarios

- **Escenario Principal:** Mover + X OK
- **Escenario Error:** El sistema no está conectado

Descripción de caso de uso

- **Descripción:** Movemos el eje X en positivo
- **Actores:** Usuario
- **Precondición:** Ninguna
- **Postcondición:** Se incrementa positivamente la posición del motor del eje X
- **Flujos Principales**

  1. El usuario quiere mover el eje X positivamente
  2. El sistema comprueba que existe conexión con la máquina CNC
  3. El sistema establece que la dirección en la que se va a mover la máquina es X+
  4. El sistema comienza el hilo de ejecución de la máquina en la dirección indicada anteriormente
  5. El sistema comprueba que no se ha llegado al limite del eje X
  6. El sistema aumenta el valor relativo del eje X según el incremento establecido
  7. El sistema aumenta el valor absoluto del eje X según el incremento establecido

- **Flujos Alternativos**

  2a El sistema no está conectado

  1. El sistema no envía nada a la máquina

4.2.3.11. Especificación caso de uso Mover - X

Identificación de escenarios

- **Escenario Principal:** Mover - X OK
- **Escenario Error:** El sistema no está conectado

Descripción de caso de uso

- **Descripción:** Movemos el eje X en negativo

- **Actores:** Usuario
- **Precondición:** Ninguna
- **Postcondición:** Se decrementa la posición del motor del eje X
- **Flujos Principales**

  1. El usuario quiere mover el eje X negativamente
  2. El sistema comprueba que existe conexión con la máquina CNC
  3. El sistema establece que la dirección en la que se va a mover la máquina es X-
  4. El sistema comienza el hilo de ejecución de la máquina en la dirección indicada anteriormente
  5. El sistema comprueba que no se ha llegado al limite del eje X
  6. El sistema disminuye el valor relativo del eje X según el incremento establecido
  7. El sistema disminuye el valor absoluto del eje X según el incremento establecido

- **Flujos Alternativos**

  2a El sistema no está conectado

  1. El sistema no envía nada a la máquina

4.2.3.12. Especificación caso de uso Mover + Y

Identificación de escenarios

- **Escenario Principal:** Mover + Y OK
- **Escenario Error:** El sistema no está conectado

Descripción de caso de uso

- **Descripción:** Movemos el eje Y en positivo
- **Actores:** Usuario
- **Precondición:** Ninguna
- **Postcondición:** Se incrementa positivamente la posición del motor del eje Y
- **Flujos Principales**

  1. El usuario quiere mover el eje Y positivamente
  2. El sistema comprueba que existe conexión con la máquina CNC

3. El sistema establece que la dirección en la que se va a mover la máquina es Y+
4. El sistema comienza el hilo de ejecución de la máquina en la dirección indicada anteriormente
5. El sistema comprueba que no se ha llegado al limite del eje Y
6. El sistema aumenta el valor relativo del eje Y según el incremento establecido
7. El sistema aumenta el valor absoluto del eje Y según el incremento establecido

- **Flujos Alternativos**

  2a El sistema no está conectado

  1. El sistema no envía nada a la máquina

4.2.3.13. Especificación caso de uso Mover - Y

Identificación de escenarios

- **Escenario Principal:** Mover - Y OK
- **Escenario Error:** El sistema no está conectado

Descripción de caso de uso

- **Descripción:** Movemos el eje Y en negativo
- **Actores:** Usuario
- **Precondición:** Ninguna
- **Postcondición:** Se decrementa la posición del motor del eje Y
- **Flujos Principales**

  1. El usuario quiere mover el eje Y negativamente
  2. El sistema comprueba que existe conexión con la máquina CNC
  3. El sistema establece que la dirección en la que se va a mover la Especificación caso de uso Mover - Y

  Identificación de escenarios

  - **Escenario Principal:** Mover - Y OK
  - **Escenario Error:** El sistema no está conectado

  Descripción de caso de uso

  - **Descripción:** Movemos el eje Y en negativo

- **Actores:** Usuario
- **Precondición:** Ninguna
- **Postcondición:** Se decrementa la posición del motor del eje Y
- **Flujos Principales**
  1. El usuario quiere mover el eje Y negativamente
  2. El sistema comprueba que existe conexión con la máquina CNC
  3. El sistema establece que la dirección en la que se va a mover la máquina es Y-
  4. El sistema comienza el hilo de ejecución de la máquina en la dirección indicada anteriormente
  5. El sistema comprueba que no se ha llegado al limite del eje Y
  6. El sistema disminuye el valor relativo del eje Y según el incremento establecido
  7. El sistema disminuye el valor absoluto del eje Y según el incremento establecido
- **Flujos Alternativos**

  2a El sistema no está conectado
  1. El sistema no envía nada a la máquina

1. máquina es Y-
2. El sistema comienza el hilo de ejecución de la máquina en la dirección indicada anteriormente
3. El sistema comprueba que no se ha llegado al limite del eje Y
4. El sistema disminuye el valor relativo del eje Y según el incremento establecido
5. El sistema disminuye el valor absoluto del eje Y según el incremento establecido

- **Flujos Alternativos**

  2a El sistema no está conectado
  1. El sistema no envía nada a la máquina

4.2.3.14. Especificación caso de uso Mover - Y

Identificación de escenarios

- **Escenario Principal:** Mover - Y OK
- **Escenario Error:** El sistema no está conectado

Descripción de caso de uso

- **Descripción:** Movemos el eje Y en negativo
- **Actores:** Usuario
- **Precondición:** Ninguna
- **Postcondición:** Se decrementa la posición del motor del eje Y
- **Flujos Principales**

  1. El usuario quiere mover el eje Y negativamente
  2. El sistema comprueba que existe conexión con la máquina CNC
  3. El sistema establece que la dirección en la que se va a mover la máquina es Y-
  4. El sistema comienza el hilo de ejecución de la máquina en la dirección indicada anteriormente
  5. El sistema comprueba que no se ha llegado al limite del eje Y
  6. El sistema disminuye el valor relativo del eje Y según el incremento establecido
  7. El sistema disminuye el valor absoluto del eje Y según el incremento establecido

- **Flujos Alternativos**

  2a El sistema no está conectado

  1. El sistema no envía nada a la máquina

4.2.3.15. Especificación caso de uso Mover + X +Y

Identificación de escenarios

- **Escenario Principal:** Mover + X + Y OK
- **Escenario Error:** El sistema no está conectado

Descripción de caso de uso

- **Descripción:** Movemos el eje X e Y en positivo
- **Actores:** Usuario
- **Precondición:** Ninguna
- **Postcondición:** Se incrementa la posición del motor del eje X e Y
- **Flujos Principales**

  1. El usuario quiere mover el eje X y Y positivamente
  2. El sistema comprueba que existe conexión con la máquina CNC

3. El sistema establece que la dirección en la que se va a mover la máquina es X+ Y+
4. El sistema comienza el hilo de ejecución de la máquina en la dirección indicada anteriormente
5. El sistema comprueba que no se ha llegado al limite del eje X e Y
6. El sistema incrementa el valor relativo del eje X según el incremento establecido
7. El sistema incrementa el valor absoluto del eje X según el incremento establecido
8. El sistema incrementa el valor relativo del eje Y según el incremento establecido
9. El sistema incrementa el valor absoluto del eje Y según el incremento establecido

- **Flujos Alternativos**

2a El sistema no está conectado

1. El sistema no envía nada a la máquina

4.2.3.16. Especificación caso de uso Mover + X -Y

Identificación de escenarios

- **Escenario Principal:** Mover + X - Y OK
- **Escenario Error:** El sistema no está conectado

Descripción de caso de uso

- **Descripción:** Movemos el eje X en positivo e Y en negativo
- **Actores:** Usuario
- **Precondición:** Ninguna
- **Postcondición:** Se incrementa la posición del motor del eje X y se decrementa el eje Y
- **Flujos Principales**

1. El usuario quiere mover el eje X positivamente y el eje Y negativamente
2. El sistema comprueba que existe conexión con la máquina CNC
3. El sistema establece que la dirección en la que se va a mover la máquina es X+ Y-
4. El sistema comienza el hilo de ejecución de la máquina en la dirección indicada anteriormente

5. El sistema comprueba que no se ha llegado al limite del eje X e Y
6. El sistema incrementa el valor relativo del eje X según el incremento establecido
7. El sistema incrementa el valor absoluto del eje X según el incremento establecido
8. El sistema decrementa el valor relativo del eje Y según el incremento establecido
9. El sistema decrementa el valor absoluto del eje Y según el incremento establecido

- **Flujos Alternativos**

2a El sistema no está conectado

1. El sistema no envía nada a la máquina

4.2.3.17. Especificación caso de uso Mover - X +Y

Identificación de escenarios

- **Escenario Principal:** Mover - X + Y OK
- **Escenario Error:** El sistema no está conectado

Descripción de caso de uso

- **Descripción:** Movemos el eje X en negativo e Y en positivo
- **Actores:** Usuario
- **Precondición:** Ninguna
- **Postcondición:** Se decrementa la posición del motor del eje X y se incrementa el eje Y
- **Flujos Principales**

1. El usuario quiere mover el eje X positivamente y el eje Y negativamente
2. El sistema comprueba que existe conexión con la máquina CNC
3. El sistema establece que la dirección en la que se va a mover la máquina es X- Y+
4. El sistema comienza el hilo de ejecución de la máquina en la dirección indicada anteriormente
5. El sistema comprueba que no se ha llegado al limite del eje X e Y
6. El sistema decrementa el valor relativo del eje X según el incremento establecido

7. El sistema decrementa el valor absoluto del eje X según el incremento establecido
8. El sistema incrementa el valor relativo del eje Y según el incremento establecido
9. El sistema incrementa el valor absoluto del eje Y según el incremento establecido

- **Flujos Alternativos**

2a El sistema no está conectado

1. El sistema no envía nada a la máquina

4.2.3.18. Especificación caso de uso Mover - X - Y

Identificación de escenarios

- **Escenario Principal:** Mover - X - Y OK
- **Escenario Error:** El sistema no está conectado

Descripción de caso de uso

- **Descripción:** Movemos el eje X e Y en negativo
- **Actores:** Usuario
- **Precondición:** Ninguna
- **Postcondición:** Se decrementa la posición del motor del eje X e Y
- **Flujos Principales**

1. El usuario quiere mover el eje X y Y negativamente
2. El sistema comprueba que existe conexión con la máquina CNC
3. El sistema establece que la dirección en la que se va a mover la máquina es X- Y-
4. El sistema comienza el hilo de ejecución de la máquina en la dirección indicada anteriormente
5. El sistema comprueba que no se ha llegado al limite del eje X e Y
6. El sistema decrementa el valor relativo del eje X según el incremento establecido
7. El sistema decrementa el valor absoluto del eje X según el incremento establecido
8. El sistema decrementa el valor relativo del eje Y según el incremento establecido

9. El sistema decrementa el valor absoluto del eje Y según el incremento establecido

- **Flujos Alternativos**

  2a El sistema no está conectado

  1. El sistema no envía nada a la máquina

4.2.3.19. Especificación caso de uso Mover + Z

Identificación de escenarios

- **Escenario Principal:** Mover + Z OK
- **Escenario Error:** El sistema no está conectado

Descripción de caso de uso

- **Descripción:** Movemos el eje Z en positivo
- **Actores:** Usuario
- **Precondición:** Ninguna
- **Postcondición:** Se incrementa positivamente la posición del motor del eje Z
- **Flujos Principales**

  1. El usuario quiere mover el eje Z positivamente
  2. El sistema comprueba que existe conexión con la máquina CNC
  3. El sistema establece que la dirección en la que se va a mover la máquina es Z+
  4. El sistema comienza el hilo de ejecución de la máquina en la dirección indicada anteriormente
  5. El sistema comprueba que no se ha llegado al limite del eje Z
  6. El sistema aumenta el valor relativo del eje Z según el incremento establecido
  7. El sistema aumenta el valor absoluto del eje Z según el incremento establecido

- **Flujos Alternativos**

  2a El sistema no está conectado

  1. El sistema no envía nada a la máquina

4.2.3.20. Especificación caso de uso Mover - Z

Identificación de escenarios

- **Escenario Principal:** Mover - Z OK
- **Escenario Error:** El sistema no está conectado

Descripción de caso de uso

- **Descripción:** Movemos el eje Z en negativo
- **Actores:** Usuario
- **Precondición:** Ninguna
- **Postcondición:** Se decrementa la posición del motor del eje Z
- **Flujos Principales**

  1. El usuario quiere mover el eje Z negativamente
  2. El sistema comprueba que existe conexión con la máquina CNC
  3. El sistema establece que la dirección en la que se va a mover la máquina es Z-
  4. El sistema comienza el hilo de ejecución de la máquina en la dirección indicada anteriormente
  5. El sistema comprueba que no se ha llegado al limite del eje Z
  6. El sistema disminuye el valor relativo del eje Z según el incremento establecido
  7. El sistema disminuye el valor absoluto del eje Z según el incremento establecido

- **Flujos Alternativos**

  2a El sistema no está conectado

  1. El sistema no envía nada a la máquina

4.2.4. Diagrama de casos de uso : Códigos G

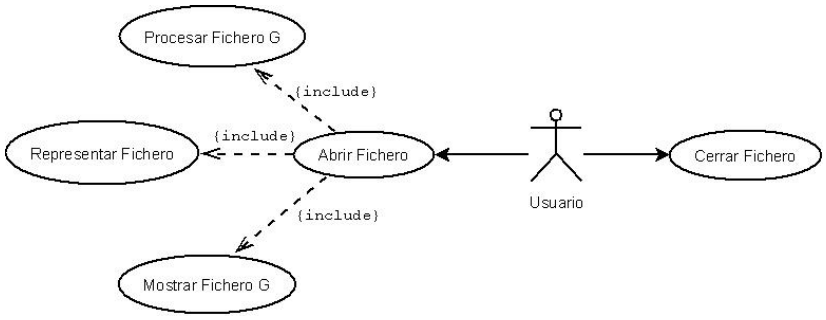

Figura 4.4: Diagrama de caso de uso : Códigos G

4.2.4.1. Especificación caso de uso Abrir Fichero

Identificación de escenarios

- **Escenario Principal:** Abrir Fichero OK
- **Escenario Error:** No se puede leer el fichero
- **Escenario Excepción:** El usuario cancela la apertura del fichero

Descripción de caso de uso

- **Descripción:** El sistema abre y filtra un fichero cnc
- **Actores:** Usuario
- **Precondición:** Ninguna
- **Postcondición:** Se abre y filtra el fichero, mostrando ambos
- **Flujos Principales**

  1. El usuario quiere abrir un fichero
  2. El sistema comprueba que si existe un fichero previamente abierto y lo cierra
  3. El sistema abre el dialogo de apertura de fichero
  4. El usuario elige el fichero cnc a abrir
  5. El sistema comprueba que se ha elegido un fichero cnc y guarda su nombre
  6. El sistema abre el fichero para solo lectura
  7. El sistema limpia el visor de fichero y configura la fuente
  8. El sistema muestra el fichero

9. El sistema comienza a recorrer el fichero para procesarlo, *Include Procesar Fichero G*
10. El sistema muestra el fichero filtrado, *Include Mostrar Fichero G*

- **Flujos Alternativos**

  5a El fichero no puede abrirse

  1. El sistema emite error

4.2.4.2. Especificación caso de uso Procesar Fichero G Nivel: Subfunción

Identificación de escenarios

- **Escenario Principal:** Procesar Fichero G OK
- **Escenario Error:**

Descripción de caso de uso

- **Descripción:** El sistema Procesa un fichero con códigos G
- **Actores:** Usuario
- **Precondición:** Se ha abierto el fichero con códigos G
- **Postcondición:** Se filtra el contenido del fichero dado con éxito
- **Flujos Principales**

  1. El sistema crea una lista para almacenar las instrucciones validas
  2. El sistema comprueba que la linea comienza por 'N'
  3. El sistema comprueba que la linea contiene una instrucción M3, o M9
  4. El sistema añade la linea a la lista, y la lista al vector de listas

- **Flujos Alternativos**

  3a El sistema comprueba que la linea contiene una instrucción G90

  1. El sistema añade G90 a la lista, la lista al vector de listas y saltamos el paso 4

  3b El sistema comprueba que la linea contiene una instrucción G91

  1. El sistema añade G91 a la lista, la lista al vector de listas y saltamos el paso 4

  3c El sistema comprueba que la linea contiene una instrucción G00, G1, G01, G3 o G03

1. El sistema crea una linea cuyos elementos son la división de la linea original en pequeñas instrucciones
2. El sistema comprueba que la linea tiene alguna instrucción válida
3. El sistema comprueba que la linea contiene X
   - 3a La linea no contiene X
     - 1 El sistema inserta '$' en la posición de la X
     - 2 El sistema comprueba que la linea contiene Y
       - 2a La linea no contiene Y
         - 1 El sistema inserta '$' en la posición de la Y
         - 2 El sistema comprueba que la linea contiene Z
           - 2a La linea no contiene Z
             - 1 El sistema inserta '$' en la posición de la Z
         - 3 El sistema introduce en la lista el valor de Z
     - 3 El sistema introduce en la lista el valor de Y
       - 1 El sistema comprueba que la linea contiene Z
         - 2a La linea no contiene Z
           - 1 El sistema inserta '$' en la posición de la Z
       - 2 El sistema introduce en la lista el valor de Z
4. El sistema introduce en la lista el valor de X
5. El sistema comprueba que la linea contiene Y
   - 2a La linea no contiene Y
     - 1 El sistema inserta '$' en la posición de la Y
     - 2 El sistema comprueba que la linea contiene Z
       - 2a La linea no contiene Z
         - 1 El sistema inserta '$' en la posición de la Z
     - 3 El sistema introduce en la lista el valor de Z
6. El sistema introduce en la lista el valor de Y
   - 1 El sistema comprueba que la linea contiene Z
     - 2a La linea no contiene Z
       - 1 El sistema inserta '$' en la posición de la Z
   - 2 El sistema introduce en la lista el valor de Z
7. El sistema comprueba que la linea contiene G00X, G00Y o G00Z
   - 1 El sistema introduce en la lista 'G00'
   - 2 El sistema introduce la lista en el vector
   - 7a El sistema comprueba que la linea contiene G1X, G01X, G1Y, G01Y, G01Z o G1Z

1 El sistema introduce en la lista 'G1'

2 El sistema comprueba que la linea contiene F y la introduce en la lista

2a El sistema inserta '$' en la posición de la F

3 El sistema introduce la lista en el vector

7b El sistema comprueba que la linea contiene G03 o G3

1 El sistema introduce en la lista 'G3'

2 El sistema comprueba que la linea contiene R y la introduce en la lista

2a El sistema inserta '$' en la posición de la R

3 El sistema comprueba que la linea contiene J y la introduce en la lista

3a El sistema inserta '$' en la posición de la J

4 El sistema comprueba que la linea contiene I y la introduce en la lista

4a El sistema inserta '$' en la posición de la I

5 El sistema introduce la lista en el vector

4.2.4.3. Especificación caso de uso Mostrar Fichero G Nivel: Subfunción

Identificación de escenarios

- **Escenario Principal:** Mostrar Fichero G OK
- **Escenario Error:**

Descripción de caso de uso

- **Descripción:** El sistema Muestra el fichero con códigos G
- **Actores:** Usuario
- **Precondición:** Se ha abierto el fichero con códigos G
- **Postcondición:** Se Muestra el contenido del fichero filtrado
- **Flujos Principales**

1. Se crea un vector de listas de cadenas con las instrucciones filtradas
2. Se crea una cadena s vacía
3. El sistema comprueba que la lista contiene una instrucción M03 y asigna a la cadena s "M03"

4. El sistema comprueba que la cadena s no está vacía y muestra el fichero filtrado por pantalla

5. Repetir pasos 2-4 hasta que no queden listas de cadenas en el vector que mostrar

- **Flujos Alternativos**

3a El sistema comprueba que la lista contiene una instrucción G90, cambia el sistema de medición como absoluto y asigna a la cadena s G90

3b El sistema comprueba que la lista contiene una instrucción G91 cambia la medición a relativo y asigna a la cadena s G91

3c El sistema comprueba que la lista contiene una instrucción M30 y asigna a la cadena s M30

3d El sistema comprueba que la lista contiene una instrucción M09 y asigna a la cadena s M09

3e El sistema comprueba que la lista contiene una instrucción G00 y asigna a la cadena s G00

1. El sistema comprueba que el segundo elemento de la lista no es nulo y concatena la cadena s con su valor X

2. El sistema comprueba que el tercer elemento de la lista no es nulo y concatena la cadena s con su valor Y

3. El sistema comprueba que el cuarto elemento de la lista no es nulo y concatena la cadena s con su valor Z

3f El sistema comprueba que la lista contiene una instrucción G1 y asigna a la cadena s G1

1. El sistema comprueba que el segundo elemento de la lista no es nulo y concatena la cadena s con su valor X

2. El sistema comprueba que el tercer elemento de la lista no es nulo y concatena la cadena s con su valor Y

3. El sistema comprueba que el cuarto elemento de la lista no es nulo y concatena la cadena s con su valor Z

4. El sistema comprueba que el quinto elemento de la lista no es nulo y concatena la cadena s con su valor F

3g El sistema comprueba que la lista contiene una instrucción G3 y asigna a la cadena s G3

1. El sistema comprueba que el segundo elemento de la lista no es nulo y concatena la cadena s con su valor X
2. El sistema comprueba que el tercer elemento de la lista no es nulo y concatena la cadena s con su valor Y
3. El sistema comprueba que el cuarto elemento de la lista no es nulo y concatena la cadena s con su valor Z
4. El sistema comprueba que el quinto elemento de la lista no es nulo y concatena la cadena s con su valor R
5. El sistema comprueba que el quinto elemento de la lista no es nulo y concatena la cadena s con su valor I
6. El sistema comprueba que el quinto elemento de la lista no es nulo y concatena la cadena s con su valor J

4.2.4.4. Especificación caso de uso Representar Fichero Nivel: Subfunción

Identificación de escenarios

- **Escenario Principal:** Representar fichero OK

Descripción de caso de uso

- **Descripción:** Se representa el fichero mediante un gráfico
- **Actores:** Usuario
- **Precondición:** El fichero está abierto y procesado
- **Postcondición:** El sistema representa el fichero a través de un gráfico
- **Flujos Principales**

1. El sistema establece la matriz de proyección.
2. El sistema inicializa la matriz identidad.
3. El sistema multiplica la matriz actual por una matriz de translación que mueve el objeto.
4. El sistema multiplica la matriz actual por una matriz que cambia el tamaño del objeto a lo largo de los ejes.
5. El sistema establece las posibles transformaciones de la cámara (perspectivas) en los 3 ejes x, y, z.
6. El sistema establece que no se ha llegado al fin del fichero, y inicializa las posiciones de referencia x, y, z.

7. El sistema comienza a recorrer el fichero que contienen las instrucciones a representar.
8. El sistema comprueba que la instrucción contiene el comando G90 y establece el sistema de medición en forma absoluta.
9. El sistema establece el fin de fichero.
10. Repetir 7 hasta que no queden listas de cadenas en el vector que representar.

- **Flujos Alternativos**

7a El sistema comprueba que la lista contiene una instrucción G91 y establece el sistema de medición en forma relativa.

7b El sistema comprueba que la lista contiene una instrucción G00 y que no se ha llegado al final del fichero.

1. El sistema comprueba que el segundo elemento de la lista no es nulo y asigna su valor a X dependiendo del sistema de medición.
2. El sistema comprueba que el tercer elemento de la lista no es nulo y asigna su valor a Y dependiendo del sistema de medición.
3. El sistema comprueba que el cuarto elemento de la lista no es nulo y asigna su valor a Z dependiendo del sistema de medición.

7c El sistema comprueba que la lista contiene una instrucción G90 y establece el sistema de medición en forma absoluta.

7d El sistema comprueba que la lista contiene una instrucción G91 y establece el sistema de medición en forma relativa.

7e El sistema comprueba que la lista contiene una instrucción G1 y que no se ha llegado al final del fichero.

1. El sistema define 3 nuevas variables correspondientes a las coordenadas x y z y se las asigna a los valores de las coordendas iniciales.
2. El sistema comprueba que el segundo elemento de la lista no es nulo y asigna su valor a X de destino, dependiendo del sistema de medición.
3. El sistema comprueba que el tercer elemento de la lista no es nulo y asigna su valor a Y de destino, dependiendo del sistema de medición.

4. El sistema comprueba que el cuarto elemento de la lista no es nulo y asigna su valor a Z de destino, dependiendo del sistema de medición.

5. El sistema establece el color del pincel en rojo y dibuja un vertice con los puntos iniciales y de destino.

6. El sistema comprueba que no se ha llegado al final del fichero y asigna las coordenadas de destino a las iniciales

7f El sistema comprueba que la lista contiene una instrucción G90 y establece el sistema de medición en forma absoluta.

7g El sistema comprueba que la lista contiene una instrucción G91 y establece el sistema de medición en forma relativa.

7h El sistema comprueba que la lista contiene una instrucción G3 y que no se ha llegado al final del fichero.

1. El sistema define 3 nuevas variables correspondientes a las coordenadas x y z y se las asigna a los valores de las coordendas iniciales y define 3 variables para calcular el angulo asignandole 0.

2. El sistema comprueba que el segundo elemento de la lista no es nulo y asigna su valor a X de destino, dependiendo del sistema de medición.

3. El sistema comprueba que el tercer elemento de la lista no es nulo y asigna su valor a Y de destino, dependiendo del sistema de medición.

4. El sistema comprueba que el cuarto elemento de la lista no es nulo y asigna su valor a Z de destino, dependiendo del sistema de medición.

5. El sistema comprueba que el quinto elemento de la lista no es nulo y asigna su valor al radio r.

6. El sistema comprueba que el sexto elemento de la lista no es nulo y asigna su valor a rx.

7. El sistema comprueba que el séptimo elemento de la lista no es nulo y asigna su valor a ry.

5a El sistema comprueba que el quinto elemento de la lista es nulo asignamdo a r el modulo de rx ry.

8. El sistema establece los angulos alfa y beta que forman.

9. El sistema comprueba que está en el 3er o 4o cuadrante y se le suma 180 a los valores iniciales de alfa y beta.
10. El sistema comprueba que alfa es menor que beta y suma a alfa 180 grados.
11. El sistema establece los valores i y j a 0.
12. El sistema comprueba que rj es mayor a y, y asigna r a i.
13. El sistema comprueba que rj es menor a y, y asigna -r a i.
14. El sistema comprueba que ri es mayor a x, y asigna r a j.
15. El sistema comprueba que ri es menor a x, y asigna -r a j.
16. El sistema establece el color del pincel en rojo y dibuja un vertice 3d con los puntos que representan el arco a dibujar, hasta que termine el arco.
17. El sistema comprueba que no se ha llegado al final del fichero y asigna las coordenadas de destino a las iniciales.

4.2.4.5. Especificación caso de uso Cerrar Fichero

Identificación de escenarios

- **Escenario Principal:** Cerrar Fichero OK

Descripción de caso de uso

- **Descripción:** Cerramos el fichero anteriormente abierto
- **Actores:** Usuario
- **Precondición:** El fichero está abierto
- **Postcondición:** El sistema cierra el fichero cnc a petición del usuario
- **Flujos Principales**

1. El usuario quiere cerrar el fichero actualmente abierto
2. El sistema limpia el visor del fichero filtrando
3. El sistema limpia el visor del fichero cnc
4. El sistema crea un fichero temporal que hace referencia al fichero abierto
5. El sistema establece el fichero como cerrado
6. El sistema cierra el fichero

4.3. Modelo conceptual de datos

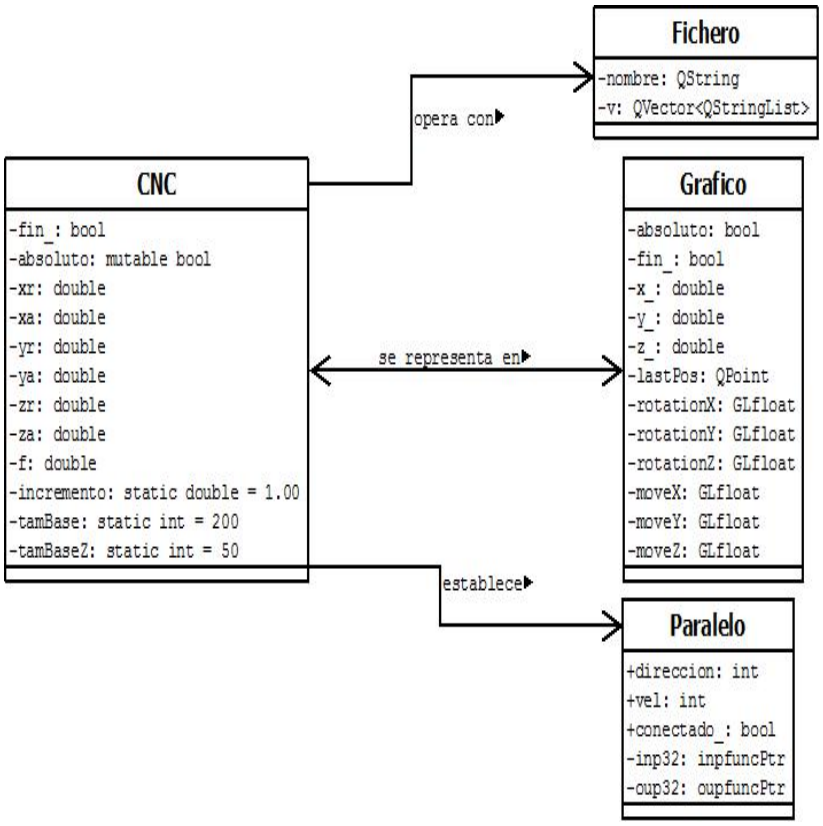

Figura 4.5: Modelo Conceptual de datos

4.4. Modelo de comportamiento del sistema

4.4.1. Escenario Principal del Caso de Uso: Ejecutar

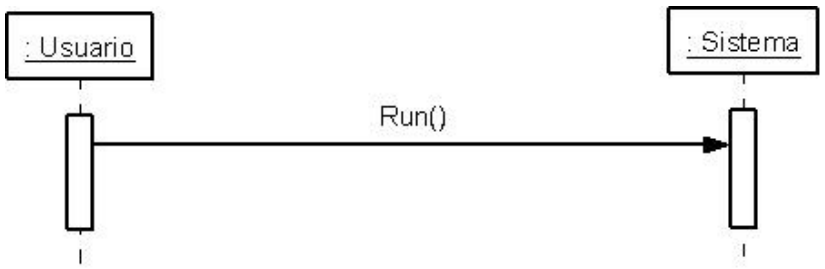

Figura 4.6: DSS: Escenario Principal cdu Ejecutar

4.4.1.1. Operación : Run

Responsabilidades: Ejecuta el fichero cnc

Referencias Cruzadas: Caso de Uso Ejecutar

Precondiciones:

- Existe una objeto CNC de la clase Cnc en curso.
- Existe una objeto F de la clase Fichero en curso.
- Existe una objeto P de la clase Paralelo en curso.

Postcondiciones:

- Se actualizó CNC.absoluto == true si el comando es G90
- Se actualizó CNC.absoluto == true si el comando es G91
- Se actualizó P.vel
- Se actualizó CNC.xa
- Se actualizó CNC.xr
- Se actualizó CNC.ya
- Se actualizó CNC.yr
- Se actualizó CNC.za
- Se actualizó CNC.zr
- Se asignó a P.dirección = nº entero [0..9], según dirección que convenga.

4.4.2. Escenario Principal del Caso de Uso: Manual

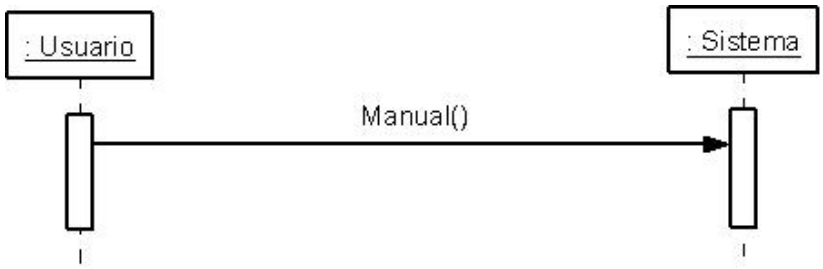

Figura 4.7: DSS: Escenario Principal cdu Manual

4.4.2.1. Operación : Manual

Responsabilidades: Muestra el manual de ayuda de la aplicación.

Referencias Cruzadas: Caso de Uso Manual

Precondiciones: Existe una objeto CNC de la clase Cnc en curso

Postcondiciones:

- Ninguna

Contrato de operaciones irrelevante

4.4.3. Escenario Principal del Caso de Uso: Ayuda

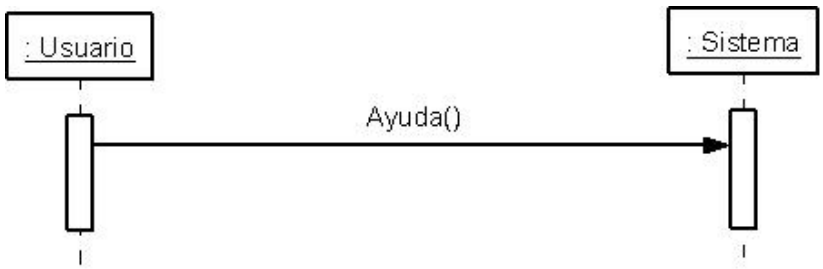

Figura 4.8: DSS: Escenario Principal cdu Ayuda

4.4.3.1. Operación : Ayuda

Responsabilidades: Muestra la ayuda acerca del sistema

Referencias Cruzadas: Caso de Uso Ayuda

Precondiciones: Existe una objeto CNC de la clase Cnc en curso

Postcondiciones:

- Ninguna

Contrato de operaciones irrelevante

4.4.4. Escenario Principal del Caso de Uso: Conectar

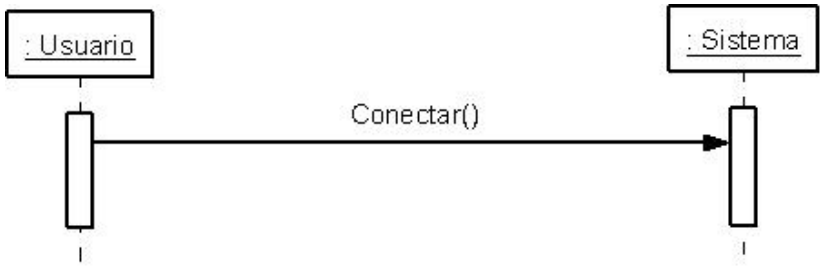

Figura 4.9: DSS: Escenario Principal cdu Conectar

4.4.4.1. Operación : Conectar

Responsabilidades: Establecer conexión con la máquina de control numérico CNC

Referencias Cruzadas: Caso de Uso Conectar

Precondiciones:

- Existe un objeto CNC de la clase Cnc en curso
- Existe un objeto P de la clase Paralelo en curso

Postcondiciones:

- Se actualizó P.vel_ según botón de velocidad pulsado
- Se actualizó a P.conectado_ = True

4.4.5. Escenario Principal del Caso de Uso: Desconectar

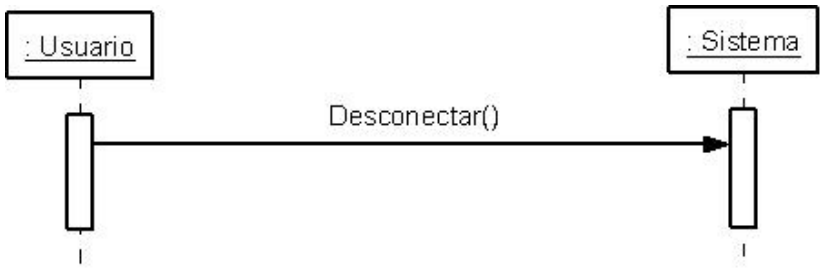

Figura 4.10: DSS: Escenario Principal cdu Desconectar

4.4.5.1. Operación : Desconectar

Responsabilidades: Establecer desconexión con la máquina de control numérico CNC

Referencias Cruzadas: Caso de Uso Desconectar

Precondiciones:

- Existe un objeto CNC de la clase Cnc en curso
- Existe un objeto P de la clase Paralelo en curso

Postcondiciones:

- Se asignó a P.conectado_ = False

4.4.6. Escenario Principal del Caso de Uso: CambiarPulso

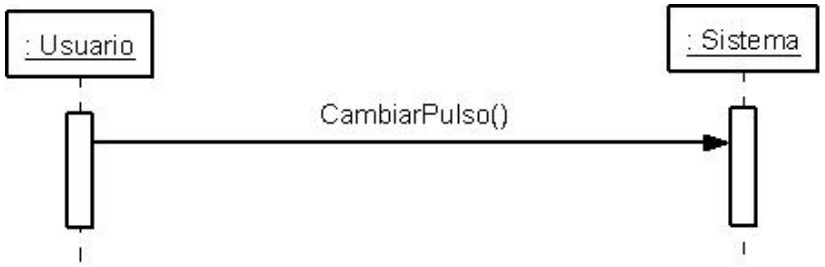

Figura 4.11: DSS: Escenario Principal cdu CambiarPulso

4.4.6.1. Operación : CambiarPulso

Responsabilidades: Establecer la anchura de pulsos a enviar a la máquina

Referencias Cruzadas: Caso de Uso CambiarPulso

Precondiciones:

- Existe un objeto CNC de la clase Cnc en curso
- Existe un objeto P de la clase Paralelo en curso

Postcondiciones:

- Se actualizó P.pulso.

4.4.7. Escenario Principal del Caso de Uso: CambiarVelpulso

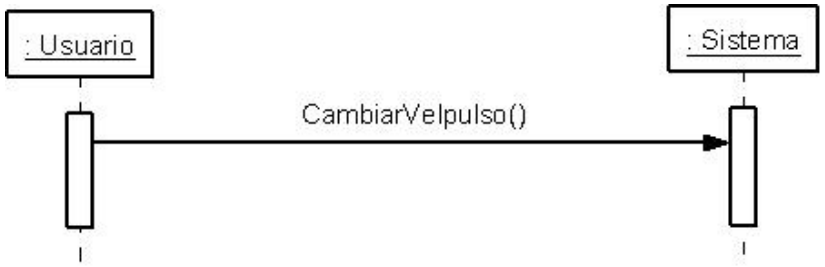

Figura 4.12: DSS: Escenario Principal cdu CambiarVelpulso

4.4.7.1. Operación : CambiarVelpulso

Responsabilidades: Establecer la velocidad de pulsos a enviar a la máquina

Referencias Cruzadas: Caso de Uso CambiarVelpulso

Precondiciones:

- Existe un objeto CNC de la clase Cnc en curso
- Existe un objeto P de la clase Paralelo en curso

Postcondiciones:

- Se actualizó P.velpulso.

4.4.8. Escenario Principal del Caso de Uso: Reset X

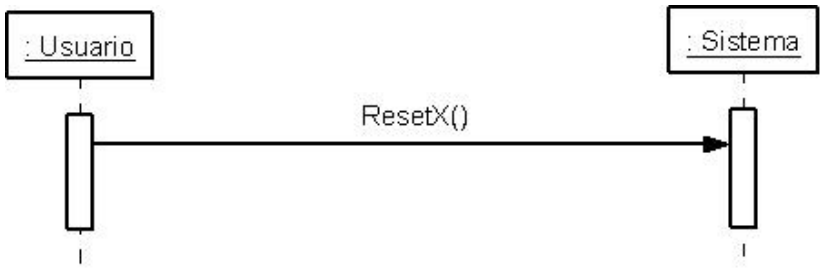

Figura 4.13: DSS: Escenario Principal cdu Reset X

4.4.8.1. Operación : Reset X

Responsabilidades: Establecer la coordenada relativa de X a 0

Referencias Cruzadas: Caso de Uso Reset X

Precondiciones:

- Existe un objeto CNC de la clase Cnc en curso

Postcondiciones:

- Se asignó a CNC.rx = 0

4.4.9. Escenario Principal del Caso de Uso: Reset Y

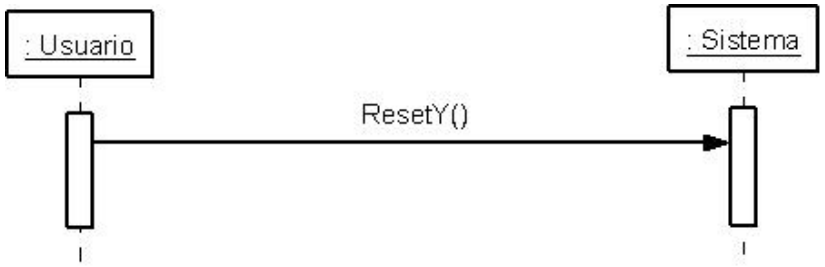

Figura 4.14: DSS: Escenario Principal cdu Reset Y

4.4.9.1. Operación : Reset Y

Responsabilidades: Establecer la coordenada relativa de Y a 0

Referencias Cruzadas: Caso de Uso Reset Y

Precondiciones:

- Existe un objeto CNC de la clase Cnc en curso

Postcondiciones:

- Se asignó a CNC.ry = 0

4.4.10. Escenario Principal del Caso de Uso: Reset Z

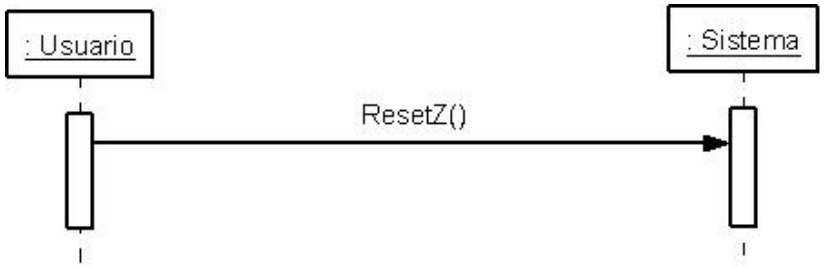

Figura 4.15: DSS: Escenario Principal cdu Reset Z

4.4.10.1. Operación : Reset Z

Responsabilidades: Establecer la coordenada relativa de Z a 0

Referencias Cruzadas: Caso de Uso Reset Z

Precondiciones:

- Existe un objeto CNC de la clase Cnc en curso

Postcondiciones:

- Se asignó a CNC.rz = 0

4.4.11. Escenario Principal del Caso de Uso: Incrementar x1

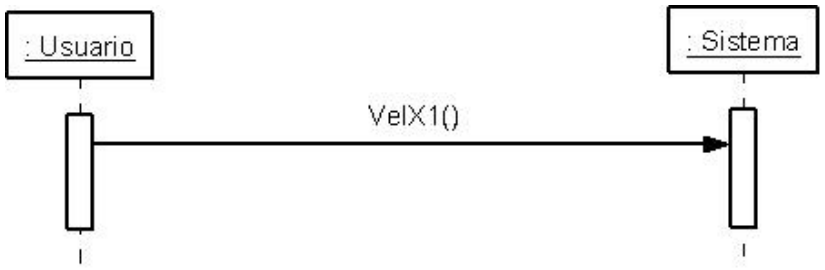

Figura 4.16: DSS: Escenario Principal cdu Incrementar x1

4.4.11.1. Operación : VelX1

Responsabilidades: Establecer el incremento x1

Referencias Cruzadas: Caso de Uso Incrementar x1

Precondiciones:

- Existe un objeto CNC de la clase Cnc en curso
- Existe un objeto P de la clase Paralelo en curso

Postcondiciones:

- Se asignó al atributo Cnc::incremento de la clase a 1.0
- Se asignó 530 al atributo vel de la clase Paralelo

4.4.12. Escenario Principal del Caso de Uso: Incrementar x2

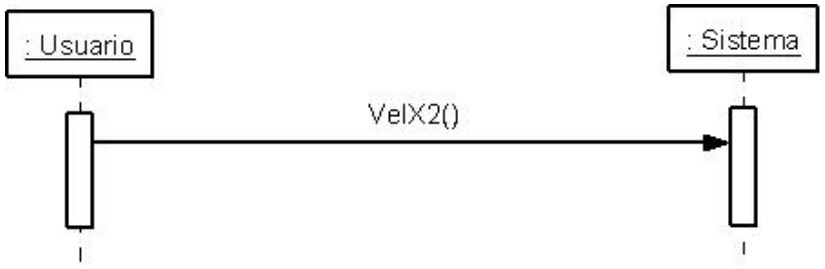

Figura 4.17: DSS: Escenario Principal cdu Incrementar x2

4.4.12.1. Operación : VelX2

Responsabilidades: Establecer el incremento x**2**

Referencias Cruzadas: Caso de Uso Incrementar x**2**

Precondiciones:

- Existe un objeto CNC de la clase Cnc en curso
- Existe un objeto P de la clase Paralelo en curso

Postcondiciones:

- Se asignó al atributo Cnc::incremento de la clase a 2.0
- Se asignó 1060 al atributo vel de la clase Paralelo

4.4.13. Escenario Principal del Caso de Uso: Incrementar x5

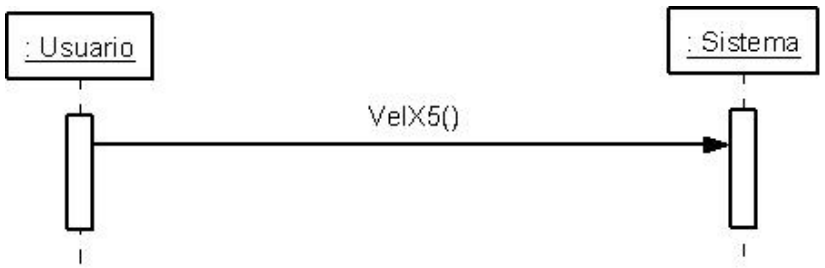

Figura 4.18: DSS: Escenario Principal cdu Incrementar x5

4.4.13.1. Operación : VelX5

Responsabilidades: Establecer el incremento x**5**

Referencias Cruzadas: Caso de Uso Incrementar X**5**

Precondiciones:

- Existe un objeto CNC de la clase Cnc en curso
- Existe un objeto P de la clase Paralelo en curso

Postcondiciones:

- Se asignó al atributo Cnc::incremento de la clase a 5.0
- Se asignó 2650 al atributo vel de la clase Paralelo

4.4.14. Escenario Principal del Caso de Uso: Incrementar x10

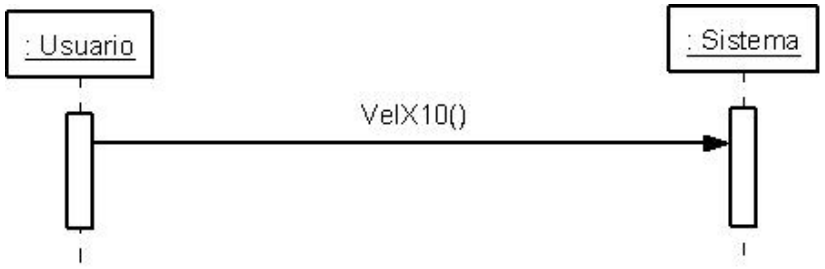

Figura 4.19: DSS: Escenario Principal cdu Incrementar x10

4.4.14.1. Operación : VelX10

Responsabilidades: Establecer el incremento x10

Referencias Cruzadas: Caso de Uso Incrementar x 10

Precondiciones:

- Existe un objeto CNC de la clase Cnc en curso
- Existe un objeto P de la clase Paralelo en curso

Postcondiciones:

- Se asignó al atributo Cnc::incremento de la clase a 10.0
- Se asignó 5300 al atributo vel de la clase Paralelo

4.4.15. Escenario Principal del Caso de Uso: Incrementar x20

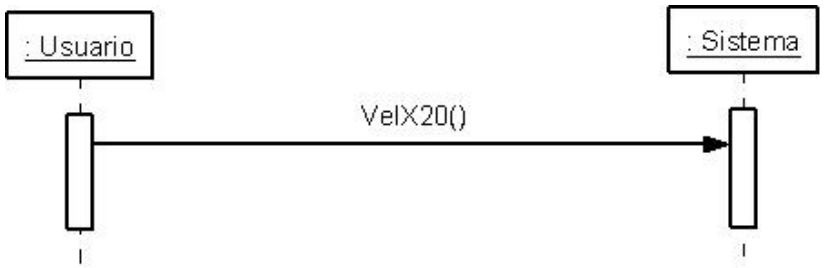

Figura 4.20: DSS: Escenario Principal cdu Incrementar x20

4.4.15.1. Operación : VelX20

Responsabilidades: Establecer el incremento x20

Referencias Cruzadas: Caso de Uso Incrementar x20

Precondiciones:

- Existe un objeto CNC de la clase Cnc en curso
- Existe un objeto P de la clase Paralelo en curso

Postcondiciones:

- Se asignó al atributo Cnc::incremento de la clase a 20.0
- Se asignó 10600 al atributo vel de la clase Paralelo

4.4.16. Escenario Principal del Caso de Uso: Incrementar x50

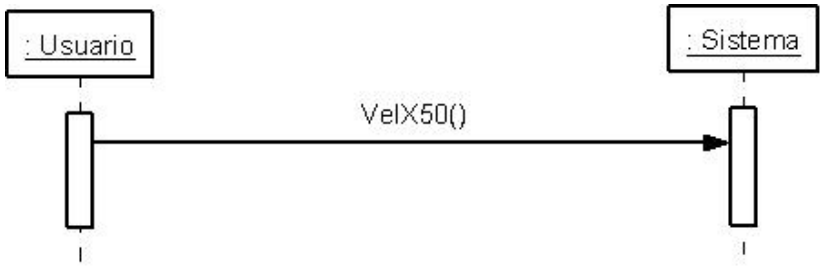

Figura 4.21: DSS: Escenario Principal cdu Incrementar x50

4.4.16.1. Operación : VelX50

Responsabilidades: Establecer el incremento x50

Referencias Cruzadas: Caso de Uso Incrementar x50

Precondiciones:

- Existe un objeto CNC de la clase Cnc en curso
- Existe un objeto P de la clase Paralelo en curso

Postcondiciones:

- Se asignó al atributo Cnc::incremento de la clase a 50.0
- Se asignó 26500 al atributo vel de la clase Paralelo

4.4.17. Escenario Principal del Caso de Uso: Mover + X

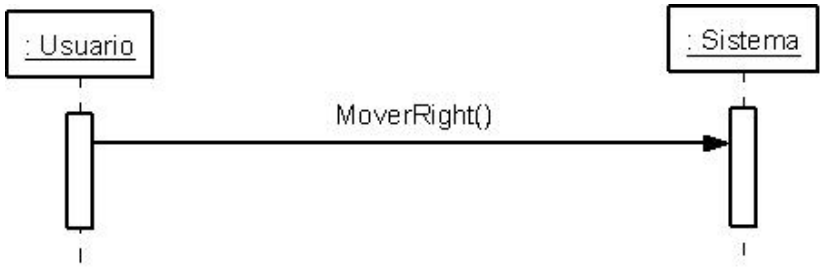

Figura 4.22: DSS: Escenario Principal cdu Mover + X

4.4.17.1. Operación : MoverRight

Responsabilidades: Incrementar X positivamente

Referencias Cruzadas: Caso de Uso Mover + X

Precondiciones:

- Existe un objeto CNC de la clase Cnc en curso
- Existe un objeto P de la clase Paralelo en curso

Postcondiciones:

- Se asignó 0 a P.dirección

4.4.18. Escenario Principal del Caso de Uso: Mover - X

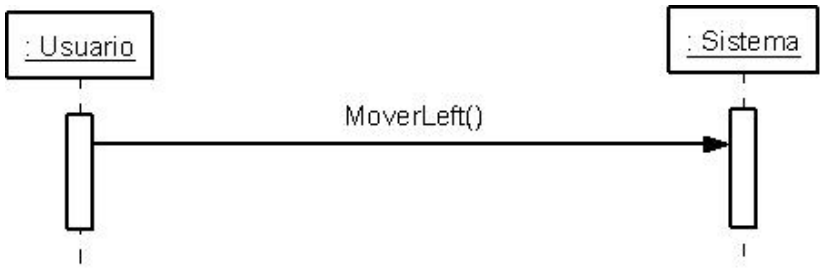

Figura 4.23: DSS: Escenario Principal cdu Mover - X

4.4.18.1. Operación : MoverLeft

Responsabilidades: Incrementar X negativamente

Referencias Cruzadas: Caso de Uso Mover - X

Precondiciones:

- Existe un objeto CNC de la clase Cnc en curso
- Existe un objeto P de la clase Paralelo en curso

Postcondiciones:

- Se asignó 1 a P.dirección

4.4.19. Escenario Principal del Caso de Uso: Mover + Y

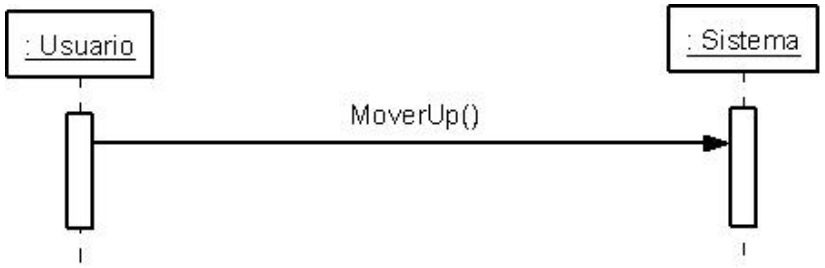

Figura 4.24: DSS: Escenario Principal cdu Mover + Y

4.4.19.1. Operación : MoverUp

Responsabilidades: Incrementar Y positivamente

Referencias Cruzadas: Caso de Uso Mover + Y

Precondiciones:

- Existe un objeto CNC de la clase Cnc en curso
- Existe un objeto P de la clase Paralelo en curso

Postcondiciones:

- Se asignó 2 a P.direccion

4.4.20. Escenario Principal del Caso de Uso: Mover - Y

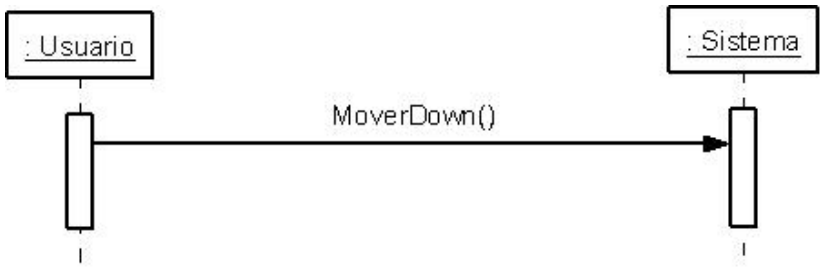

Figura 4.25: DSS: Escenario Principal cdu Mover - Y

4.4.20.1. Operación : MoverDown

Responsabilidades: Incrementar Y negativamente

Referencias Cruzadas: Caso de Uso Mover - Y

Precondiciones:

- Existe un objeto CNC de la clase Cnc en curso
- Existe un objeto P de la clase Paralelo en curso

Postcondiciones:

- Se asignó 3 a P.direccion

4.4.21. Escenario Principal del Caso de Uso: Mover + Z

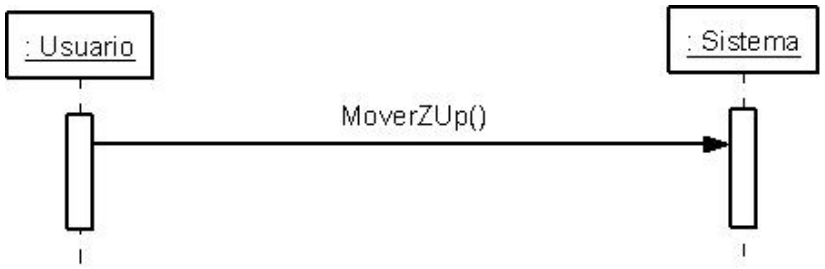

Figura 4.26: DSS: Escenario Principal cdu Mover + Z

4.4.21.1. Operación : MoverZUp

Responsabilidades: Incrementar Z positivamente

Referencias Cruzadas: Caso de Uso Mover + Z

Precondiciones:

- Existe un objeto CNC de la clase Cnc en curso
- Existe un objeto P de la clase Paralelo en curso

Postcondiciones:

- Se asignó 4 a P.direccion

4.4.22. Escenario Principal del Caso de Uso: Mover - Z

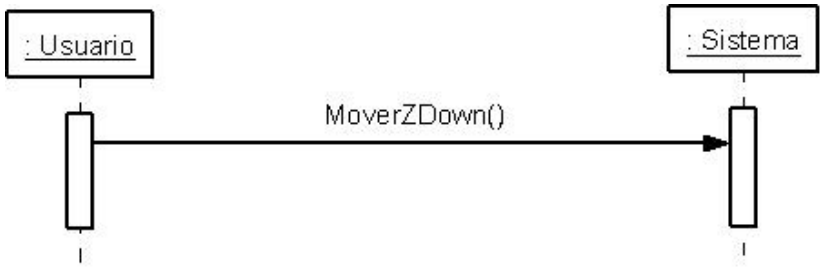

Figura 4.27: DSS: Escenario Principal cdu Mover - Z

4.4.22.1. Operación : MoverZDown

Responsabilidades: Incrementar Z negativamente

Referencias Cruzadas: Caso de Uso Mover - Z

Precondiciones:

- Existe un objeto CNC de la clase Cnc en curso
- Existe un objeto P de la clase Paralelo en curso

Postcondiciones:

- Se asignó 5 a P.direccion

4.4.23. Escenario Principal del Caso de Uso: Mover + X + Y

Figura 4.28: DSS: Escenario Principal cdu Mover + X + Y

4.4.23.1. Operación : MoverUpRight

Responsabilidades: Incrementar en diagonal X e Y positivamente

Referencias Cruzadas: Caso de Uso Mover + X + Y

Precondiciones:

- Existe un objeto CNC de la clase Cnc en curso
- Existe un objeto P de la clase Paralelo en curso

Postcondiciones:

- Se asignó 6 a P.direccion

4.4.24. Escenario Principal del Caso de Uso: Mover + X - Y

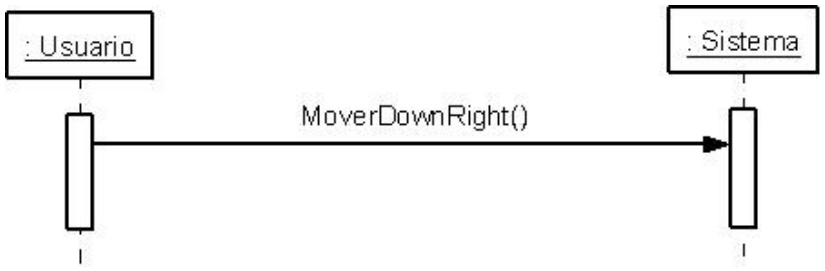

Figura 4.29: DSS: Escenario Principal cdu Mover + X - Y

4.4.24.1. Operación : MoverDownRight

Responsabilidades: Incrementar en diagonal X positivo Y negativo

Referencias Cruzadas: Caso de Uso Mover + X - Y

Precondiciones:

- Existe un objeto CNC de la clase Cnc en curso
- Existe un objeto P de la clase Paralelo en curso

Postcondiciones:

- Se asignó 7 a P.direccion

4.4.25. Escenario Principal del Caso de Uso: Mover - X + Y

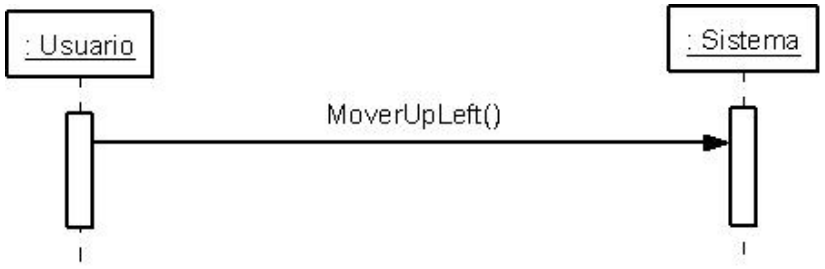

Figura 4.30: DSS: Escenario Principal cdu Mover - X + Y

4.4.25.1. Operación : MoverUpLeft

Responsabilidades: Incrementar en diagonal X negativo Y positivo

Referencias Cruzadas: Caso de Uso Mover + X + Y

Precondiciones:

- Existe un objeto CNC de la clase Cnc en curso
- Existe un objeto P de la clase Paralelo en curso

Postcondiciones:

- Se asignó 8 a P.direccion

4.4.26. Escenario Principal del Caso de Uso: Mover - X - Y

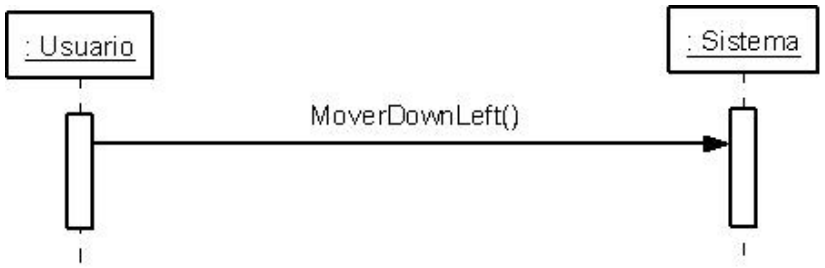

Figura 4.31: DSS: Escenario Principal cdu Mover - X - Y

4.4.26.1. Operación : MoverDownLeft

Responsabilidades: Incrementar en diagonal X e Y negativamente

Referencias Cruzadas: Caso de Uso Mover - X - Y

Precondiciones:

- Existe un objeto CNC de la clase Cnc en curso
- Existe un objeto P de la clase Paralelo en curso

Postcondiciones:

- Se asignó 9 a P.direccion

4.4.27. Escenario Principal del Caso de Uso: Abrir Fichero

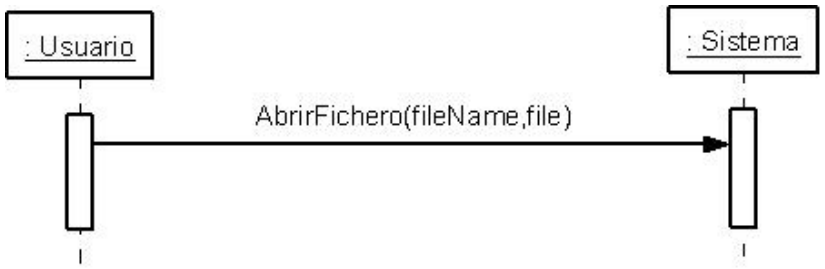

Figura 4.32: DSS: Escenario Principal cdu Abrir Fichero

4.4.27.1. Operación : AbrirFichero

Responsabilidades: Abrir fichero cnc

Referencias Cruzadas: Caso de Uso Abrir Fichero

Precondiciones:

- Existe un objeto CNC de la clase Cnc en curso

Postcondiciones:

- Se creó una instancia F de la clase Fichero y se asignó
  - F.nombre = fileName
- Se asignó el atributo file a F.file
- Se asignó true a CNC.abierto

4.4.28. Escenario Principal del Caso de Uso: Cerrar Fichero

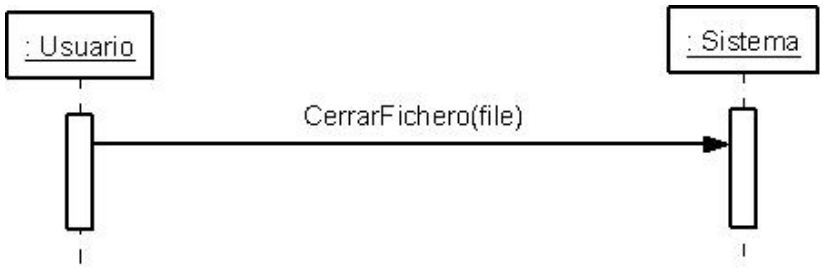

Figura 4.33: DSS: Escenario Principal cdu Cerrar Fichero

4.4.28.1. Operación : CerrarFichero

Responsabilidades: Cerrar fichero cnc

Referencias Cruzadas: Caso de Uso Cerrar Fichero

Precondiciones:

- Existe un objeto CNC de la clase Cnc en curso
- Existe un objeto F de la clase Fichero en curso

Postcondiciones:

- Se asignó el atributo f a F.file
- Se asignó false a CNC.abierto

4.4.29. Escenario Principal del Caso de Uso: Procesar Fichero G

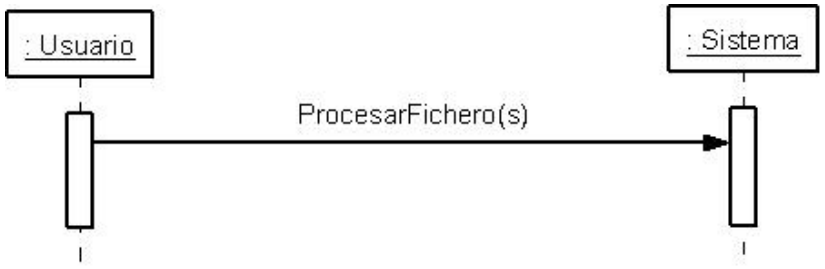

Figura 4.34: DSS: Escenario Principal cdu Procesar Fichero G

4.4.29.1. Operación : ProcesarFichero

Responsabilidades: Procesar un fichero cnc

Referencias Cruzadas: Caso de Uso Procesar Fichero G

Precondiciones:

- Existe un objeto CNC de la clase Cnc en curso
- Existe un objeto F de la clase Fichero en curso

Postcondiciones:

- Se actualizó el vector de instrucciones F.v.

4.4.30. Escenario Principal del Caso de Uso: Mostrar Fichero G

Figura 4.35: DSS: Escenario Principal cdu Mostrar Fichero G

4.4.30.1. Operación : MostrarFichero

Responsabilidades: Mostrar fichero cnc

Referencias Cruzadas: Caso de Uso Mostrar Fichero G

Precondiciones:

- Existe un objeto CNC de la clase Cnc en curso
- Existe un objeto F de la clase Fichero en curso

Postcondiciones:

- Se comprueba si el comando es G90 o G91
- Se actualizó CNC.absoluto == true si el comando es G90
- Se actualizó CNC.absoluto == true si el comando es G91

4.4.31. Escenario Principal del Caso de Uso: Representar Fichero

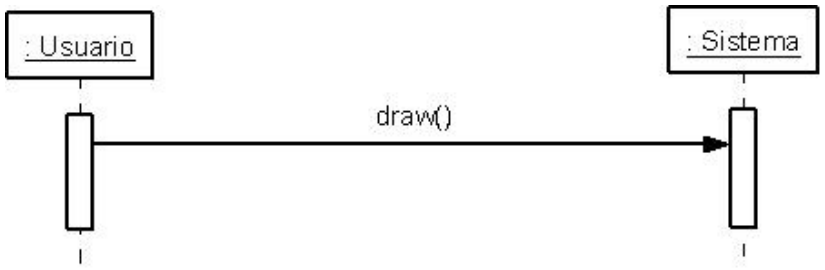

Figura 4.36: DSS: Escenario Principal cdu Representar Fichero

4.4.31.1. Operación : draw()

Responsabilidades: Representar el gráfico constituido por el fichero G

Referencias Cruzadas: Caso de Uso Representar Fichero

Precondiciones:

- Existe un objeto CNC de la clase Cnc en curso
- Existe un objeto F de la clase Fichero en curso
- Existe un objeto G de la clase Gráfico en curso

Postcondiciones:

- Se actualizó G.x_
- Se actualizó G.y_
- Se actualizó G.z_
- Se comprueba si el comando es G90 o G91
- Se actualizó CNC.absoluto == true si el comando es G90
- Se actualizó CNC.absoluto == true si el comando es G91
- Se actualizó G.fin_ con true o false

Capítulo 5

Diseño

5.1. Diagrama de clases de diseño

En el diagrama de clases de diseño se ha representado las clases y atributos más relevantes para el diseño, omitiéndose en este caso todo lo relacionado con la interfaz gráfica de Qt.

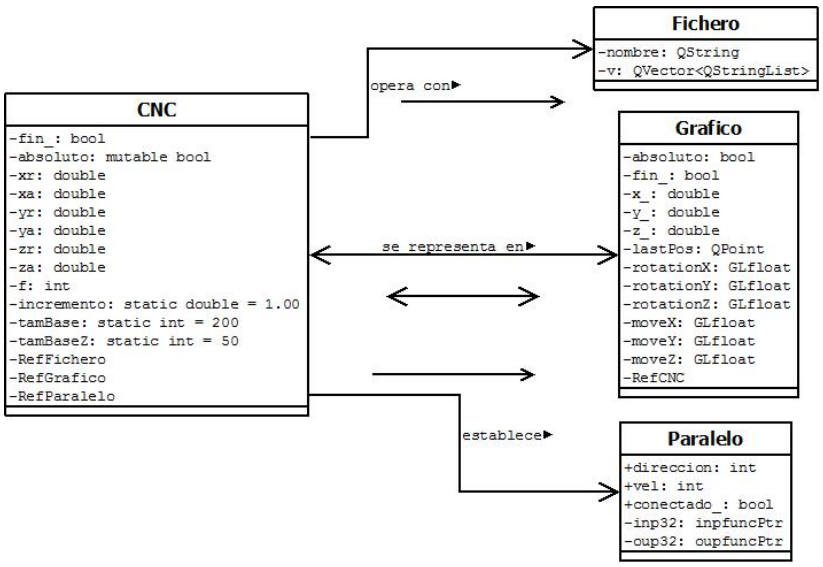

Figura 5.1: Diagrama de clases de diseño

5.2. Diagrama de iteración

5.2.1. Diagrama de secuencia

En el diagrama de secuencia mostramos la interacción del conjunto de objetos de la aplicación a través del tiempo y modelamos esto para cada caso de uso.

5.2.1.1. Escenario principal del caso de uso Conectar

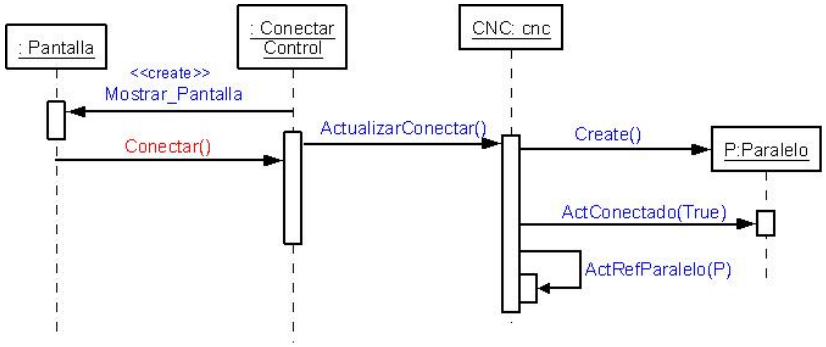

Figura 5.2: DS: Escenario principal del caso de uso Conectar

5.2.1.2. Escenario principal del caso de uso CambiarPulso

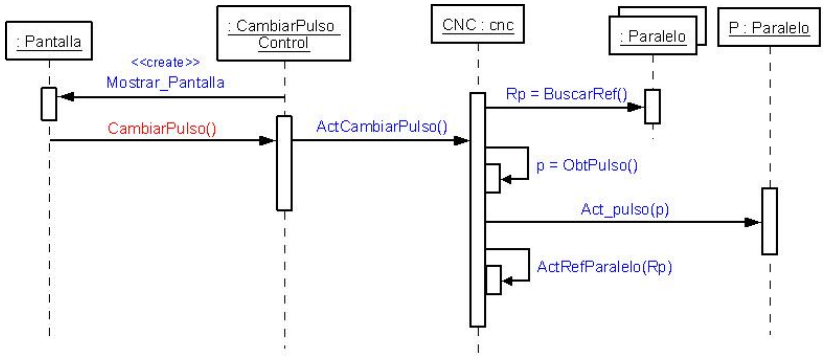

Figura 5.3: DS: Escenario principal del caso de uso CambiarPulso

5.2.1.3. Escenario principal del caso de uso CambiarVelpulso

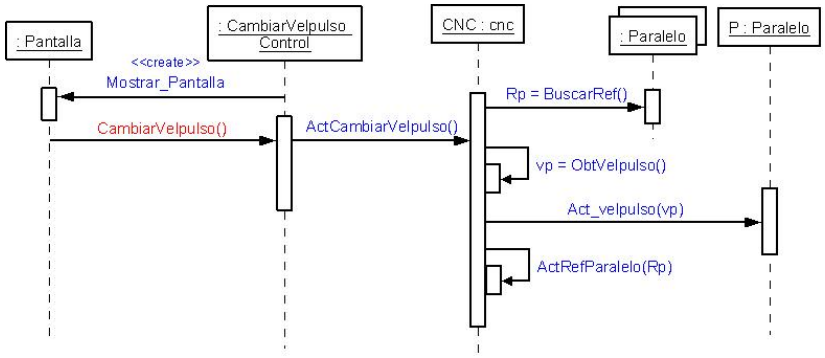

Figura 5.4: DS: Escenario principal del caso de uso CambiarVelpulso

5.2.1.4. Escenario principal del caso de uso Desconectar

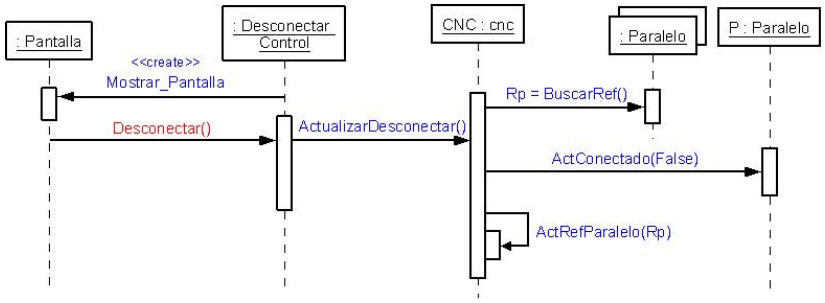

Figura 5.5: DS: Escenario principal del caso de uso Desconectar

5.2.1.5. Escenario principal del caso de uso Reset X

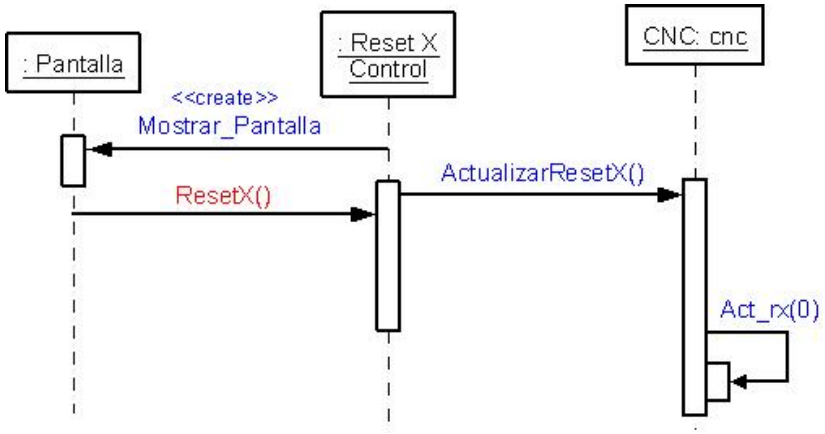

Figura 5.6: DS: Escenario principal del caso de uso Reset X

5.2.1.6. Escenario principal del caso de uso Reset Y

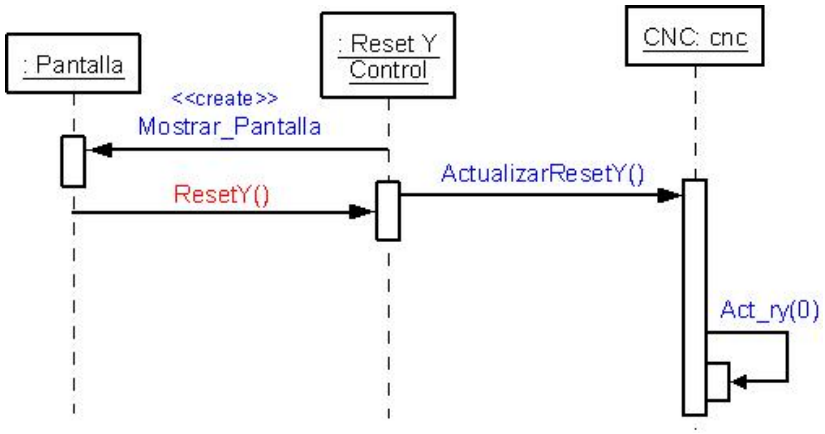

Figura 5.7: DS: Escenario principal del caso de uso Reset Y

5.2.1.7. Escenario principal del caso de uso Reset Z

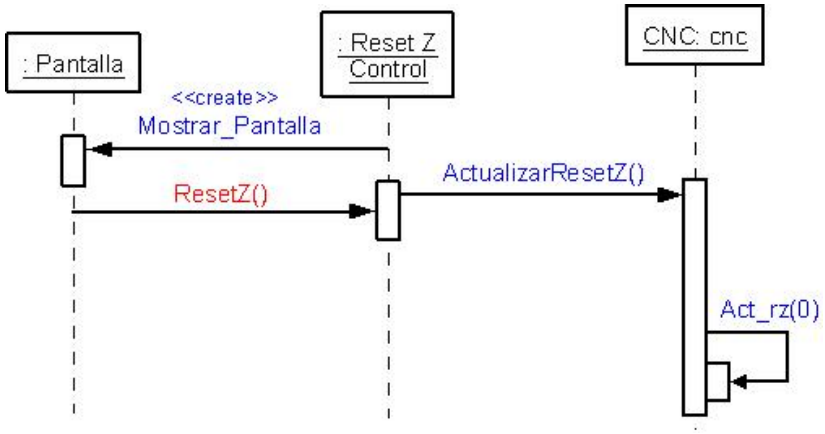

Figura 5.8: DS: Escenario principal del caso de uso Reset Z

5.2.1.8. Escenario principal del caso de uso Incrementar x1

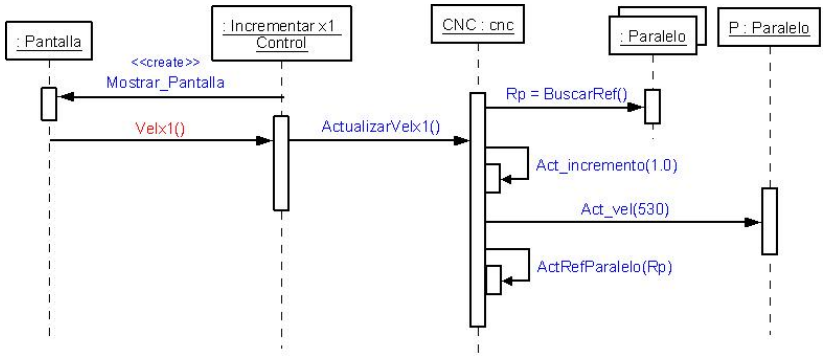

Figura 5.9: DS: Escenario principal del caso de uso Incrementar x1

5.2.1.9. Escenario principal del caso de uso Incrementar x2

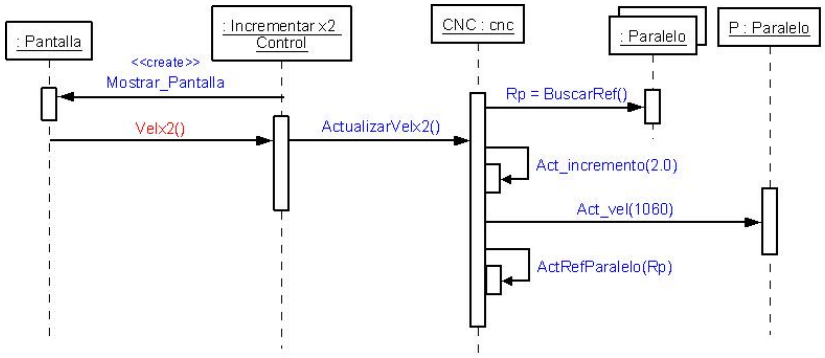

Figura 5.10: DS: Escenario principal del caso de uso Incrementar x2

5.2.1.10. Escenario principal del caso de uso Incrementar x5

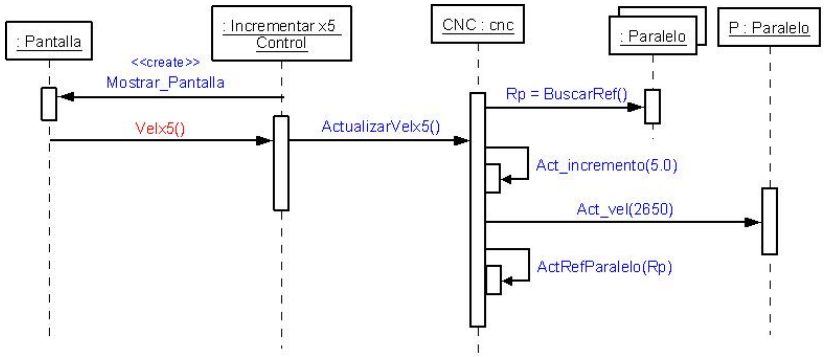

Figura 5.11: DS: Escenario principal del caso de uso Incrementar x5

5.2.1.11. Escenario principal del caso de uso Incrementar x10

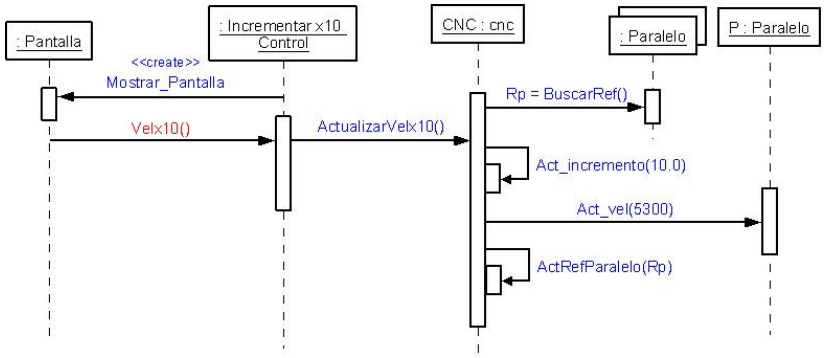

Figura 5.12: DS: Escenario principal del caso de uso Incrementar x10

5.2.1.12. Escenario principal del caso de uso Incrementar x20

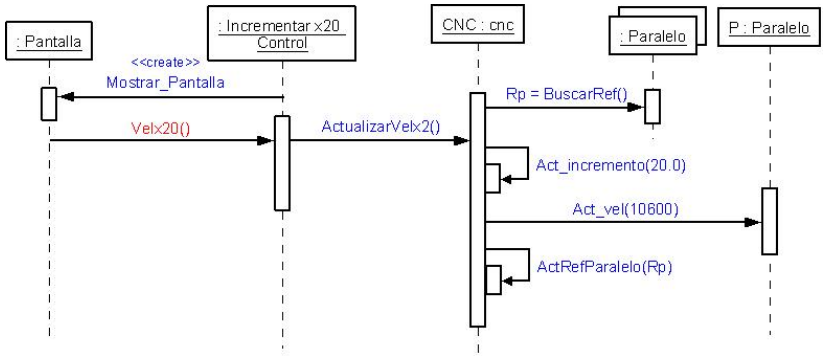

Figura 5.13: DS: Escenario principal del caso de uso Incrementar x20

5.2.1.13. Escenario principal del caso de uso Incrementar x50

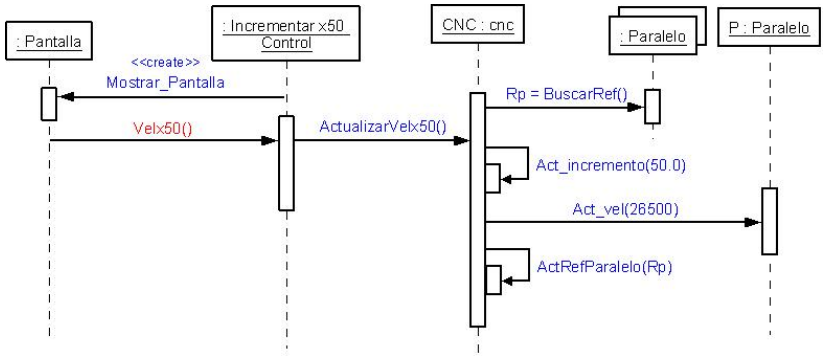

Figura 5.14: DS: Escenario principal del caso de uso Incrementar x50

5.2.1.14. Escenario principal del caso de uso Mover + X

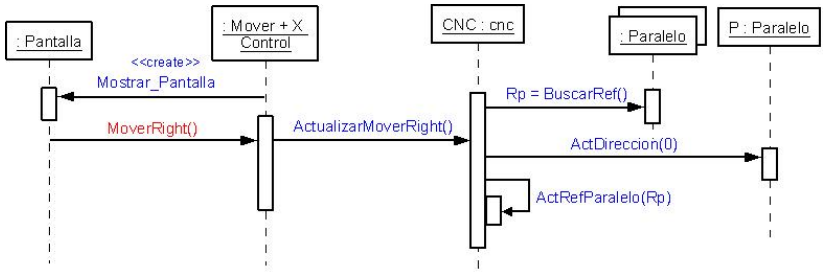

Figura 5.15: DS: Escenario principal del caso de uso Mover + X

5.2.1.15. Escenario principal del caso de uso Mover - X

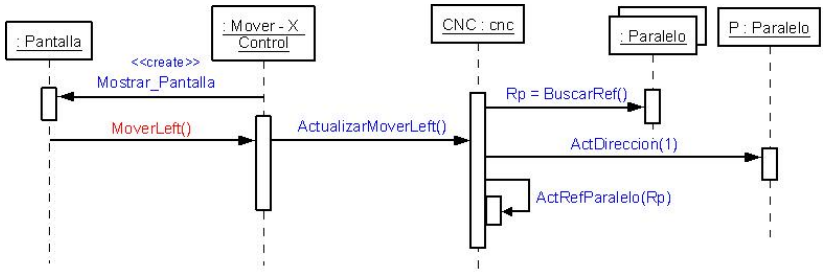

Figura 5.16: DS: Escenario principal del caso de uso Mover - X

5.2.1.16. Escenario principal del caso de uso Mover + Y

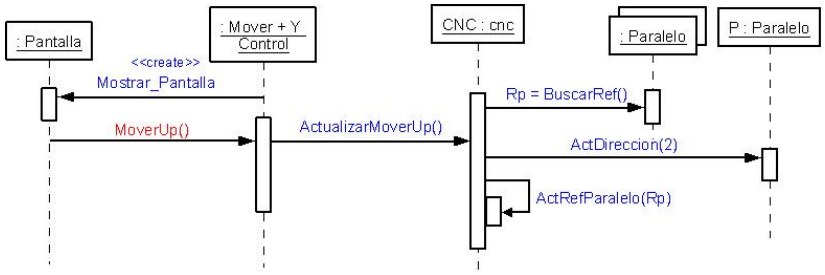

Figura 5.17: DS: Escenario principal del caso de uso Mover + Y

5.2.1.17. Escenario principal del caso de uso Mover - Y

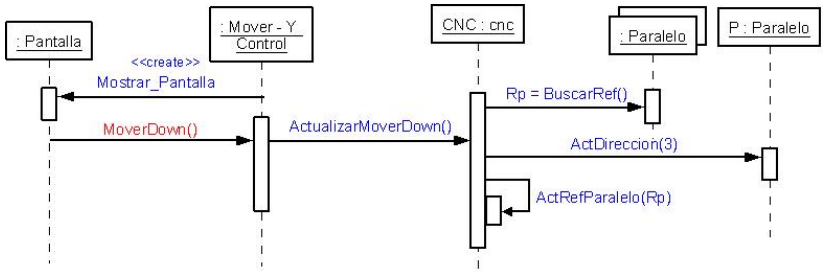

Figura 5.18: DS: Escenario principal del caso de uso Mover - Y

5.2.1.18. Escenario principal del caso de uso Mover + Z

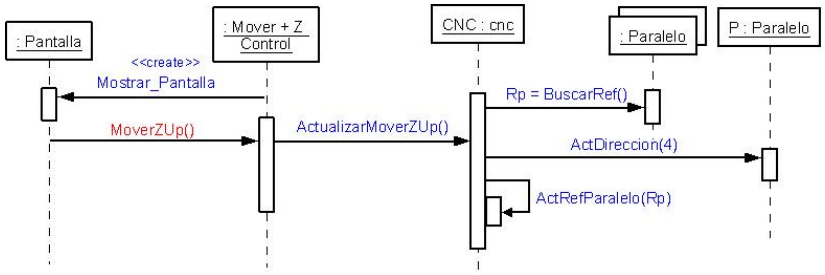

Figura 5.19: DS: Escenario principal del caso de uso Mover + Z

5.2.1.19. Escenario principal del caso de uso Mover - Z

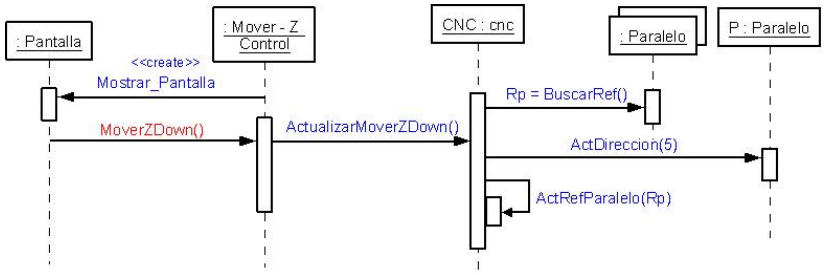

Figura 5.20: DS: Escenario principal del caso de uso Mover - Z

5.2.1.20. Escenario principal del caso de uso Mover - Z

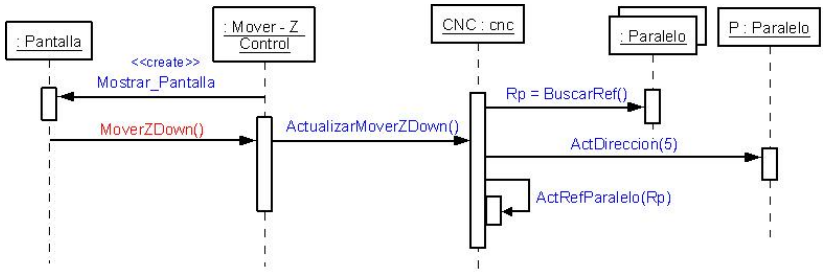

Figura 5.21: DS: Escenario principal del caso de uso Mover - Z

5.2.1.21. Escenario principal del caso de uso Mover + X + Y

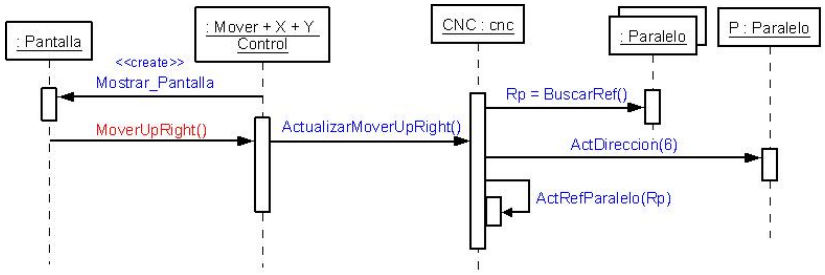

Figura 5.22: DS: Escenario principal del caso de uso Mover + X + Y

5.2.1.22. Escenario principal del caso de uso Mover + X - Y

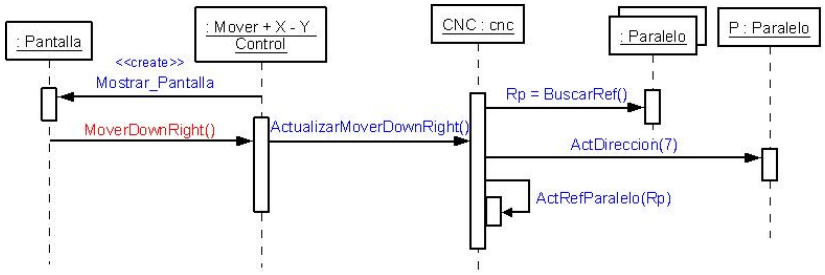

Figura 5.23: DS: Escenario principal del caso de uso Mover + X - Y

5.2.1.23. Escenario principal del caso de uso Mover - X + Y

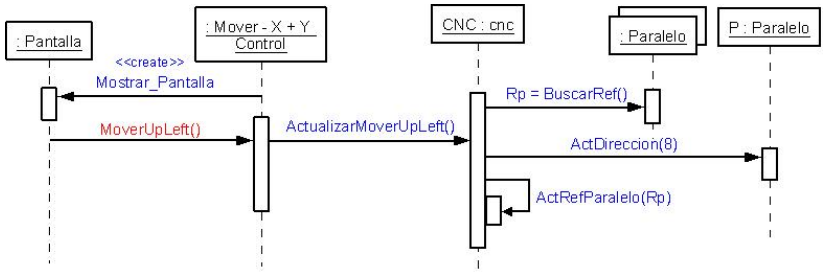

Figura 5.24: DS: Escenario principal del caso de uso Mover - X + Y

5.2.1.24. Escenario principal del caso de uso Mover - X - Y

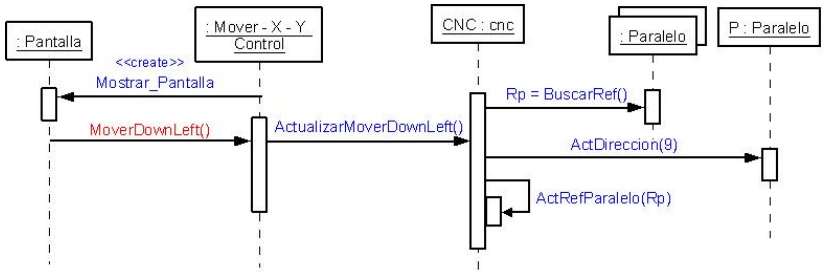

Figura 5.25: DS: Escenario principal del caso de uso Mover - X - Y

5.2.1.25. Escenario principal del caso de uso Abrir Fichero

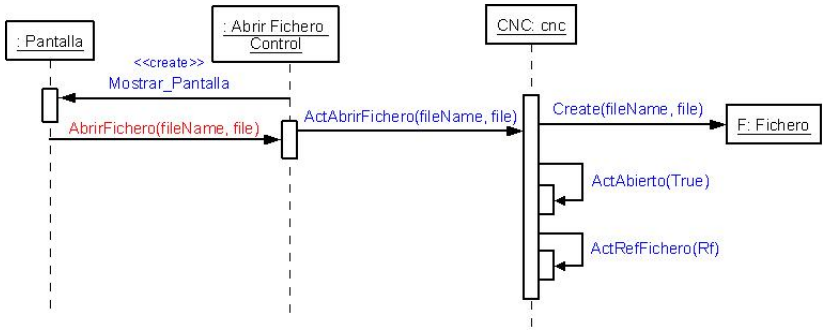

Figura 5.26: DS: Escenario principal del caso de uso Abrir Fichero

5.2.1.26. Escenario principal del caso de uso Cerrar Fichero

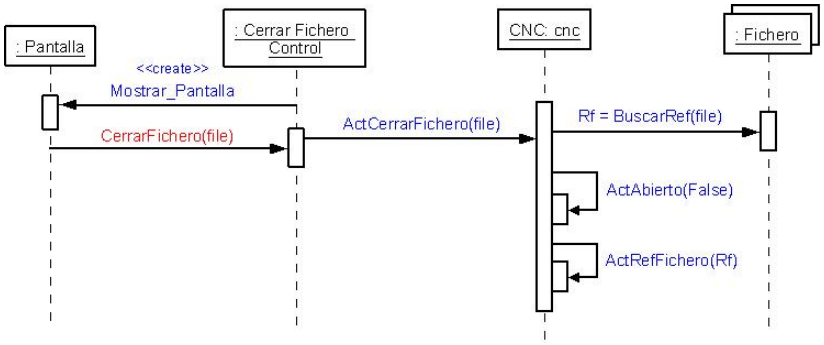

Figura 5.27: DS: Escenario principal del caso de uso Cerrar Fichero

5.2.1.27. Escenario principal del caso de uso Procesar Fichero

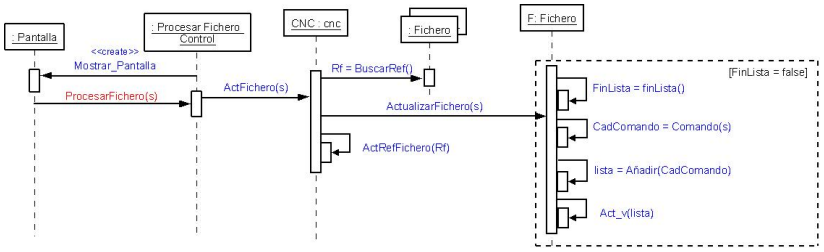

Figura 5.28: DS: Escenario principal del caso de uso Procesar Fichero

5.2.1.28. Escenario principal del caso de uso Mostrar Fichero

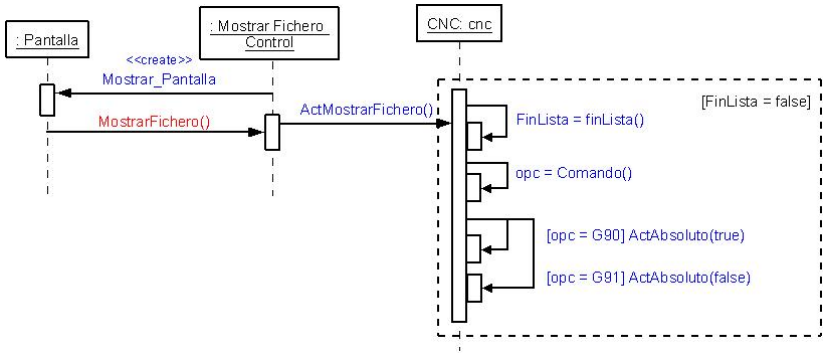

Figura 5.29: DS: Escenario principal del caso de uso Mostrar Fichero

5.2.1.29. Escenario principal del caso de uso Representar Fichero

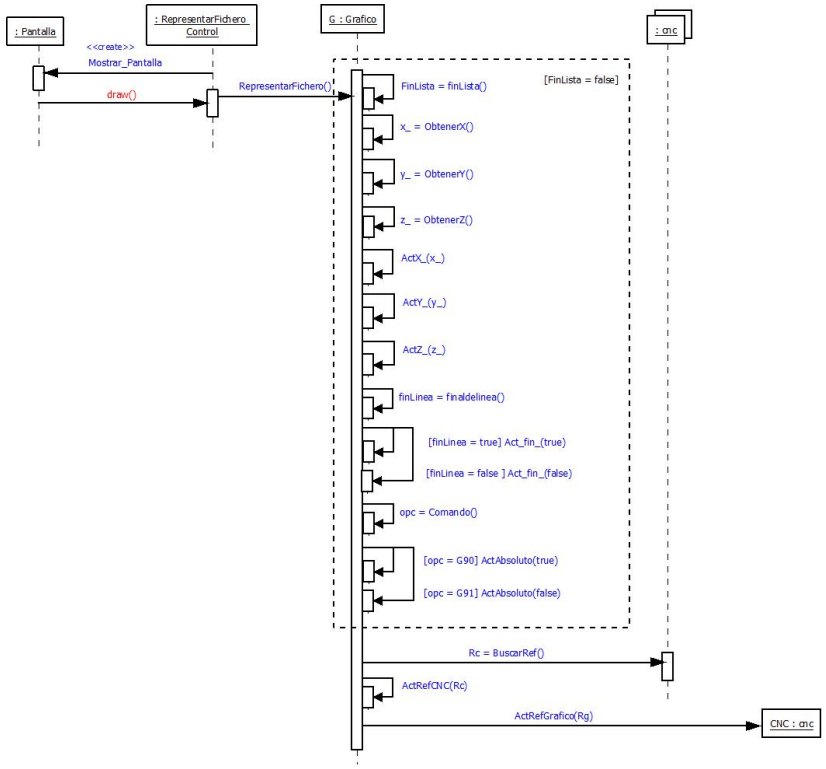

Figura 5.30: DS: Escenario principal del caso de uso Representar Fichero

5.2.1.30. Escenario principal del caso de uso Ejecutar

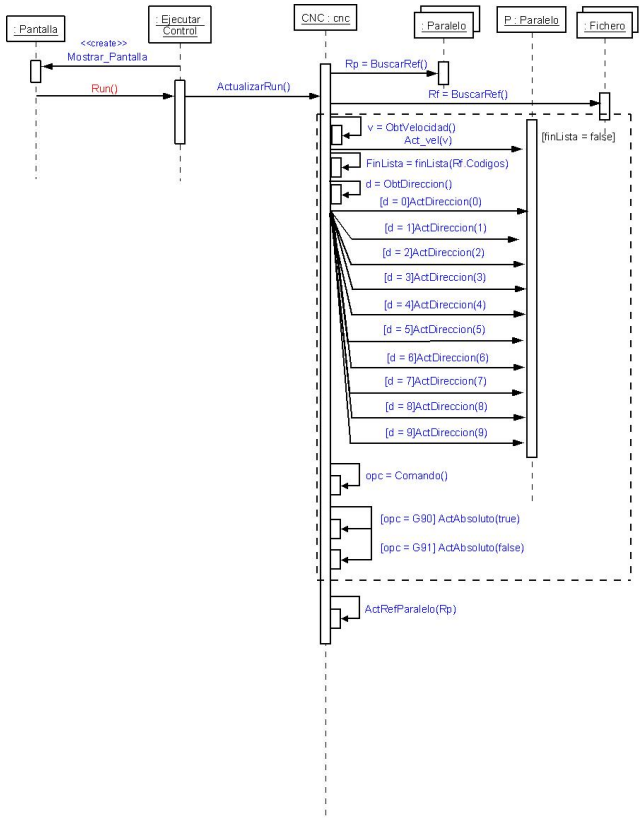

Figura 5.31: DS: Escenario principal del caso de uso Representar Fichero

5.2.2. Diagrama de clases de diseño

5.2.2.1. Clases

- **Clases nuevas**
  - Pantalla
  - Conectar Control
  - Desconectar Control
  - CambiarPulso Control
  - CambiarVelPulso Control
  - Reset X Control

- Reset Y Control
- Reset Z Control
- Incrementar x1 Control
- Incrementar x2 Control
- Incrementar x5 Control
- Incrementar x10 Control
- Incrementar x20 Control
- Incrementar x50 Control
- Mover + X Control
- Mover - X Control
- Mover + Y Control
- Mover - Y Control
- Mover + Z Control
- Mover - Z Control
- Mover + X + Y Control
- Mover + X - Y Control
- Mover - X + Y Control
- Mover - X - Y Control
- Abrir Fichero Control
- Cerrar Fichero Control
- Procesar Fichero Control
- Mostrar Fichero Control
- Representar Fichero Control
- Ejecutar Control

5.2.2.2. Asociaciones

- Colaboraciones entre objetos
    - Pantalla y Conectar Control
    - Pantalla y Desconectar Control
    - Pantalla y CambiarPulso Control
    - Pantalla y CambiarVelpulso Control
    - Pantalla y Reset X Control
    - Pantalla y Reset Y Control
    - Pantalla y Reset Z Control
    - Pantalla y Incrementar x1 Control
    - Pantalla y Incrementar x2 Control
    - Pantalla y Incrementar x5 Control
    - Pantalla y Incrementar x10 Control
    - Pantalla y Incrementar x20 Control
    - Pantalla y Incrementar x50 Control
    - Pantalla y Mover + X Control
    - Pantalla y Mover - X Control
    - Pantalla y Mover + Y Control
    - Pantalla y Mover - Y Control
    - Pantalla y Mover + Z Control
    - Pantalla y Mover - Z Control
    - Pantalla y Mover + X + Y Control
    - Pantalla y Mover + X - Y Control
    - Pantalla y Mover - X + Y Control
    - Pantalla y Mover - X - Y Control
    - Pantalla y Abrir Fichero Control
    - Pantalla y Cerrar Fichero Control
    - Pantalla y Procesar Fichero Control

- Pantalla y Representar Fichero Control
- Pantalla y Ejecutar Control
- Conectar Control y CNC
- Desconectar Control y CNC
- CambiarPulso Control y CNC
- CambiarVelpulso Control y CNC
- CNC y Paralelo
- Reset X Control y CNC
- Reset Y Control y CNC
- Reset Z Control y CNC
- Incrementar x1 Control y CNC
- Incrementar x2 Control y CNC
- Incrementar x5 Control y CNC
- Incrementar x10 Control y CNC
- Incrementar x20 Control y CNC
- Incrementar x50 Control y CNC
- Mover + X Control y CNC
- Mover - X Control y CNC
- Mover + Y Control y CNC
- Mover - Y Control y CNC
- Mover + Z Control y CNC
- Mover - Z Control y CNC
- Mover + X + Y Control y CNC
- Mover + X - Y Control y CNC
- Mover - X + Y Control y CNC
- Mover - X - Y Control y CNC
- Abrir Fichero Control y CNC
- Cerrar Fichero Control y CNC

- Procesar Fichero Control y CNC
- Mostrar Fichero Control y Gráfico
- Representar Fichero Control y Gráfico
- Ejecutar Control y Cnc
- CNC y Fichero
- Gráfico y Cnc

5.2.2.3. Operaciones

- **Clase Pantalla**
  - **Operación Mostrar Pantalla**. Esta operación muestra la pantalla del menú principal de la aplicación.
- **Clase Conectar Control**
  - **Operación que se ejecuta al hacer new.ConectarControl:** Invoca a la operación Mostrar Pantalla de la clase Pantalla.
  - **Operación Conectar():** Invoca a la operación ActualizarConectar de la clase CNC.
- **Clase Desconectar Control**
  - **Operación que se ejecuta al hacer new.DesconectarControl:** Invoca a la operación Mostrar Pantalla de la clase Pantalla.
  - **Operación Desconectar():** Invoca a la operación ActualizarDesconectar de la clase CNC.
- **Clase CambiarPulso Control**
  - **Operación que se ejecuta al hacer new.CambiarPulsoControl:** Invoca a la operación Mostrar Pantalla de la clase Pantalla.
  - **Operación CambiarPulso():** Invoca a la operación ActCambiarPulso de la clase CNC.
- **Clase CambiarVelpulso Control**
  - **Operación que se ejecuta al hacer new.CambiarVelpulsoControl:** Invoca a la operación Mostrar Pantalla de la clase Pantalla.

- **Operación CambiarVelpulso():** Invoca a la operación ActCambiarVelpulso de la clase CNC.

- **Clase Reset X Control**

  - **Operación que se ejecuta al hacer new.ResetXControl:** Invoca a la operación Mostrar Pantalla de la clase Pantalla.
  - **Operación ResetX():** Invoca a la operación ActualizarResetX() de la clase CNC.

- **Clase Reset Y Control**

  - **Operación que se ejecuta al hacer new.ResetYControl:** Invoca a la operación Mostrar Pantalla de la clase Pantalla.
  - **Operación ResetZ():** Invoca a la operación ActualizarResetY() de la clase CNC.

- **Clase Reset Z Control**

  - **Operación que se ejecuta al hacer new.ResetZControl:** Invoca a la operación Mostrar Pantalla de la clase Pantalla.
  - **Operación ResetZ():** Invoca a la operación ActualizarResetZ() de la clase CNC.

- **Clase Incrementar x1 Control**

  - **Operación que se ejecuta al hacer new.Incrementarx1Control:** Invoca a la operación Mostrar Pantalla de la clase Pantalla.
  - **Operación Velx1():** Invoca a la operación ActualizarVelx1() de la clase CNC.

- **Clase Incrementar x2 Control**

  - **Operación que se ejecuta al hacer new.Incrementarx2Control:** Invoca a la operación Mostrar Pantalla de la clase Pantalla.
  - **Operación Velx2():** Invoca a la operación ActualizarVelx2() de la clase CNC.

- **Clase Incrementar x5 Control**

  - **Operación que se ejecuta al hacer new.Incrementarx5Control:** Invoca a la operación Mostrar Pantalla de la clase Pantalla.

- **Operación Velx5():** Invoca a la operación ActualizarVelx5() de la clase CNC.

- **Clase Incrementar x10 Control**

  - **Operación que se ejecuta al hacer new.Incrementarx10Control:** Invoca a la operación Mostrar Pantalla de la clase Pantalla.
  - **Operación Velx10():** Invoca a la operación ActualizarVelx10() de la clase CNC.

- **Clase Incrementar x20 Control**

  - **Operación que se ejecuta al hacer new.Incrementarx20Control:** Invoca a la operación Mostrar Pantalla de la clase Pantalla.
  - **Operación Velx20():** Invoca a la operación ActualizarVelx20() de la clase CNC.

- **Clase Incrementar x50 Control**

  - **Operación que se ejecuta al hacer new.Incrementarx50Control:** Invoca a la operación Mostrar Pantalla de la clase Pantalla.
  - **Operación Velx50():** Invoca a la operación ActualizarVelx50() de la clase CNC.

- **Clase Mover + X Control**

  - **Operación que se ejecuta al hacer new.Mover+XControl:** Invoca a la operación Mostrar Pantalla de la clase Pantalla.
  - **Operación MoverRight():** Invoca a la operación ActualizarMoverRight() de la clase CNC.

- **Clase Mover - X Control**

  - **Operación que se ejecuta al hacer new.Mover-XControl:** Invoca a la operación Mostrar Pantalla de la clase Pantalla.
  - **Operación MoverLeft():** Invoca a la operación ActualizarMoverLeft() de la clase CNC.

- **Clase Mover + Y Control**

  - **Operación que se ejecuta al hacer new.Mover+YControl:** Invoca a la operación Mostrar Pantalla de la clase Pantalla.

- **Operación MoverUp():** Invoca a la operación ActualizarMoverUp() de la clase CNC.

- **Clase Mover -Y Control**

  - **Operación que se ejecuta al hacer new.Mover-YControl:** Invoca a la operación Mostrar Pantalla de la clase Pantalla.
  - **Operación MoverDown():** Invoca a la operación ActualizarMoverDown() de la clase CNC.

- **Clase Mover + Z Control**

  - **Operación que se ejecuta al hacer new.Mover+ZControl:** Invoca a la operación Mostrar Pantalla de la clase Pantalla.
  - **Operación MoverZUp():** Invoca a la operación ActualizarMoverZUp() de la clase CNC.

- **Clase Mover - Z Control**

  - **Operación que se ejecuta al hacer new.Mover-ZControl:** Invoca a la operación Mostrar Pantalla de la clase Pantalla.
  - **Operación MoverZDown():** Invoca a la operación ActualizarZDown() de la clase CNC.

- **Clase Mover + X + Y Control**

  - **Operación que se ejecuta al hacer new.Mover+X+YControl:** Invoca a la operación Mostrar Pantalla de la clase Pantalla.
  - **Operación MoverUpRight():** Invoca a la operación ActualizarUpRight() de la clase CNC.

- **Clase Mover + X - Y Control**

  - **Operación que se ejecuta al hacer new.Mover+X-YControl:** Invoca a la operación Mostrar Pantalla de la clase Pantalla.
  - **Operación MoverDownRight():** Invoca a la operación ActualizarDownRight() de la clase CNC.

- **Clase Mover - X + Y Control**

  - **Operación que se ejecuta al hacer new.Mover-X+YControl:** Invoca a la operación Mostrar Pantalla de la clase Pantalla.

- **Operación MoverUpLeft():** Invoca a la operación ActualizarUpLeft() de la clase CNC.

- **Clase Mover - X - Y Control**

  - **Operación que se ejecuta al hacer new.Mover-X-YControl:** Invoca a la operación Mostrar Pantalla de la clase Pantalla.
  - **Operación MoverDownLeft():** Invoca a la operación ActualizarDownLeft() de la clase CNC.

- **Clase Abrir Fichero Control**

  - **Operación que se ejecuta al hacer new.AbrirFicheroControl:** Invoca a la operación Mostrar Pantalla de la clase Pantalla.
  - **Operación AbrirFichero(fileName,file):** Invoca a la operación ActualizarAbrirFichero(fileName,file) de la clase CNC.

- **Clase Cerrar Fichero Control**

  - **Operación que se ejecuta al hacer new.CerrarFicheroControl:** Invoca a la operación Mostrar Pantalla de la clase Pantalla.
  - **Operación CerrarFichero(file):** Invoca a la operación ActualizarCerrarFichero(file) de la clase CNC.

- **Clase Procesar Fichero Control**

  - **Operación que se ejecuta al hacer new.ProcesarFicheroControl:** Invoca a la operación Mostrar Pantalla de la clase Pantalla.
  - **Operación ProcesarFichero(s):** Invoca a la operación ActualizarFichero(s) de la clase CNC.

- **Clase Mostrar Fichero Control**

  - **Operación que se ejecuta al hacer new.MostrarFicheroControl:** Invoca a la operación Mostrar Pantalla de la clase Pantalla.
  - **Operación MostrarFichero():** Invoca a la operación ActMostrarFichero() de la clase CNC.

- **Clase Representar Fichero Control**

  - **Operación que se ejecuta al hacer new.RepresentarFicheroControl:** Invoca a la operación Mostrar Pantalla de la clase Pantalla.

- **Operación draw():** Invoca a la operación RepresentarFichero() de la clase CNC.

- **Clase Ejecutar Control**

  - **Operación que se ejecuta al hacer new.EjecutarControl:** Invoca a la operación Mostrar Pantalla de la clase Pantalla.
  - **Operación Run():** Invoca a la operaciónActualizarRun() de la clase CNC.

- **Clase CNC**

  - **Operación ActualizarConectar():**
    - Invoca a la operación Create() de la clase Paralelo.
    - Invoca a la operación ActConectado(True) de la clase Paralelo.
    - Invoca a la operación ActRefParalelo(Rp) de la clase Paralelo.
  - **Operación ActRefParalelo(Rp):** asigna la referencia del objeto P a CNC.RefParalelo.
  - **Operación ActualizarDesconectar():**
    - Invoca a la operación Rp = BuscarRef() de la clase Paralelo.
    - Invoca a la operación ActConectado(False) de la clase Paralelo.
    - Invoca a la operación ActRefParalelo(Rp) de la clase Paralelo.
  - **Operación ActCambiarPulso():**
    - Invoca a la operación Rp = BuscarRef() de la clase Paralelo.
    - Invoca a la operación p = ObtPulso() de la clase Cnc.
    - Invoca a la operación Act_pulso(p) de la clase Paralelo.
    - Invoca a la operación ActRefParalelo(Rp) de la clase Paralelo.
  - **Operación ActCambiarVelpulso():**
    - Invoca a la operación Rp = BuscarRef() de la clase Paralelo.
    - Invoca a la operación vp = ObtVelpulso() de la clase Cnc.
    - Invoca a la operación Act_Velpulso(vp) de la clase Paralelo.

  - Invoca a la operación ActRefParalelo(Rp) de la clase Paralelo.

- **Operación p = ObtPulso():** Devuelve la anchura de pulso actual.
- **Operación vp = ObtVelpulso():** Devuelve la velocidad de pulso actual.
- **Operación ActualizarResetX():**

  - Invoca a la operación Act_rx(0) de la clase CNC.

- **Operación Act_rx(0):** CNC.xr := 0.
- **Operación ActualizarResetY():**

  - Invoca a la operación Act_ry(0) de la clase CNC.

- **Operación Act_ry(0):** CNC.yr := 0.
- **Operación ActualizarResetZ():**

  - Invoca a la operación Act_rz(0) de la clase CNC.

- **Operación Act_rz(0):** CNC.zr := 0.
- **Operación ActualizarVelx1():**

  - Invoca a la operación Rp = BuscarRef() de la clase Paralelo.
  - Invoca a la operación Act_incremento(1.0) de la clase Cnc.
  - Invoca a la operación Act_vel(530) de la clase Paralelo.
  - Invoca a la operación ActRefParalelo(Rp) de la clase Cnc.

- **Operación Act_incremento(1.0):** CNC::incremento := 1.0.
- **Operación ActualizarVelx2():**

  - Invoca a la operación Rp = BuscarRef() de la clase Paralelo.
  - Invoca a la operación Act_incremento(2.0) de la clase Cnc.
  - Invoca a la operación Act_vel(1060) de la clase Paralelo.
  - Invoca a la operación ActRefParalelo(Rp) de la clase Cnc.

- **Operación Act_incremento(2.0):** CNC::incremento := 2.0.

- **Operación ActualizarVelx5():**
  - Invoca a la operación Rp = BuscarRef() de la clase Paralelo.
  - Invoca a la operación Act_incremento(5.0) de la clase Cnc.
  - Invoca a la operación Act_vel(2650) de la clase Paralelo.
  - Invoca a la operación ActRefParalelo(Rp) de la clase Cnc.
- **Operación Act_incremento(5.0):** CNC::incremento := 5.0.
- **Operación ActualizarVelx10():**
  - Invoca a la operación Rp = BuscarRef() de la clase Paralelo.
  - Invoca a la operación Act_incremento(10.0) de la clase Cnc.
  - Invoca a la operación Act_vel(5300) de la clase Paralelo.
  - Invoca a la operación ActRefParalelo(Rp) de la clase Cnc.
- **Operación Act_incremento(10.0):** CNC::incremento := 10.0.
- **Operación ActualizarVelx20():**
  - Invoca a la operación Rp = BuscarRef() de la clase Paralelo.
  - Invoca a la operación Act_incremento(20.0) de la clase Cnc.
  - Invoca a la operación Act_vel(10600) de la clase Paralelo.
  - Invoca a la operación ActRefParalelo(Rp) de la clase Cnc.
- **Operación Act_incremento(20.0):** CNC::incremento := 20.0.
- **Operación ActualizarVelx50():**
  - Invoca a la operación Rp = BuscarRef() de la clase Paralelo.
  - Invoca a la operación Act_incremento(50.0) de la clase Cnc.
  - Invoca a la operación Act_vel(26500) de la clase Paralelo.
  - Invoca a la operación ActRefParalelo(Rp) de la clase Cnc.
- **Operación Act_incremento(50.0):** CNC::incremento := 50.0.
- **Operación ActualizarMoverRight():**
  - Invoca a la operación Rp = BuscarRef() de la clase Paralelo.

- Invoca a la operación ActDireccion(0) de la clase Paralelo.
- Invoca a la operación ActRefParalelo(Rp) de la clase Cnc.

- **Operación ActualizarMoverLeft():**
  - Invoca a la operación Rp = BuscarRef() de la clase Paralelo.
  - Invoca a la operación ActDireccion(1) de la clase Paralelo.
  - Invoca a la operación ActRefParalelo(Rp) de la clase Cnc.

- **Operación ActualizarMoverUp():**
  - Invoca a la operación Rp = BuscarRef() de la clase Paralelo.
  - Invoca a la operación ActDireccion(2) de la clase Paralelo.
  - Invoca a la operación ActRefParalelo(Rp) de la clase Cnc.

- **Operación ActualizarMoverDown():**
  - Invoca a la operación Rp = BuscarRef() de la clase Paralelo.
  - Invoca a la operación ActDireccion(3) de la clase Paralelo.
  - Invoca a la operación ActRefParalelo(Rp) de la clase Cnc.

- **Operación ActualizarMoverZUp():**
  - Invoca a la operación Rp = BuscarRef() de la clase Paralelo.
  - Invoca a la operación ActDireccion(4) de la clase Paralelo.
  - Invoca a la operación ActRefParalelo(Rp) de la clase Cnc.

- **Operación ActualizarMoverZDown():**
  - Invoca a la operación Rp = BuscarRef() de la clase Paralelo.
  - Invoca a la operación ActDireccion(5) de la clase Paralelo.
  - Invoca a la operación ActRefParalelo(Rp) de la clase Cnc.

- **Operación ActualizarMoverUpRight():**
  - Invoca a la operación Rp = BuscarRef() de la clase Paralelo.
  - Invoca a la operación ActDireccion(6) de la clase Paralelo.

- Invoca a la operación ActRefParalelo(Rp) de la clase Cnc.

- **Operación ActualizarMoverDownRight():**
  - Invoca a la operación Rp = BuscarRef() de la clase Paralelo.
  - Invoca a la operación ActDireccion(7) de la clase Paralelo.
  - Invoca a la operación ActRefParalelo(Rp) de la clase Cnc.

- **Operación ActualizarMoverUpLeft():**
  - Invoca a la operación Rp = BuscarRef() de la clase Paralelo.
  - Invoca a la operación ActDireccion(8) de la clase Paralelo.
  - Invoca a la operación ActRefParalelo(Rp) de la clase Cnc.

- **Operación ActualizarMoverDownLeft():**
  - Invoca a la operación Rp = BuscarRef() de la clase Paralelo.
  - Invoca a la operación ActDireccion(9) de la clase Paralelo.
  - Invoca a la operación ActRefParalelo(Rp) de la clase Cnc.

- **Operación ActualizarAbrirFichero(fileName, file):**
  - Invoca a la operación Create(fileName,file) de la clase Fichero.
  - Invoca a la operación ActAbierto(True) de la clase Paralelo.
  - Invoca a la operación ActRefFichero(Rf) de la clase Cnc.

- **Operación ActAbierto(True):** CNC.abierto := True.
- **Operación ActRefFichero(Rf):** asigna la referencia del objeto F a CNC.RefFichero.
- **Operación ActualizarCerrarFichero(file):**
  - Invoca a la operación Rf = BuscarRef(file) de la clase Fichero.
  - Invoca a la operación ActAbierto(False) de la clase Paralelo.
  - Invoca a la operación ActRefFichero(Rf) de la clase Cnc.

- **Operación ActAbierto(False):** CNC.abierto := False.

- **Operación ActFichero(s):**
  - Invoca a la operación Rf = BuscarRef(file) de la clase Fichero.
  - Invoca a la operación ActualizarFichero(s) de la clase Fichero.
  - Invoca a la operación ActRefFichero(Rf) de la clase Cnc.
- **Operación ActMostrarFichero():**
  - Invoca a la operación FinLista = finLista() de la clase Cnc.
  - Invoca a la operación opc = Comando() de la clase Cnc.
  - Invoca a la operación [opc = G90]ActAbsoluto(true) de la clase Cnc.
  - Invoca a la operación [opc = G91]ActAbsoluto(false) de la clase Cnc.
- **Operación FinLista = finLista():** devuelve true o false dependiendo de si ha llegado al fin de la lista o no.
- **Operación FinLista = finLista(Rf.Codigos):** devuelve true o false dependiendo de si ha llegado al fin de la lista Rf.Codigos o no.
- **Operación opc = Comando():** Devuelve el comando actual.
- **Operación d = ObtDireccion():** Devuelve la dirección actual.
- **Operación v = ObtVelocidad():** Devuelve la velocidad actual.
- **Operación [opc = G90]ActAbsoluto(true):** CNC.absoluto := true.
- **Operación [opc = G91]ActAbsoluto(false):** CNC.absoluto := false.
- **Operación Rc = BuscarRef():** Busca la referencia del objeto CNC.
- **Operación ActRefGrafico(Rg):** asigna la referencia del objeto G a CNC.RefGrafico.
- **Operación ActualizarRun():**
  - Invoca a la operación Rp = BuscarRef() de la clase Paralelo.
  - Invoca a la operación Rf = BuscarRef() de la clase Fichero
  - Invoca a la operación v = ObtVelocidad() de la clase Cnc.

- Invoca a la operación Act_vel(v) de la clase Paralelo.
- Invoca a la operación FinLista = finLista(Rf.Codigos) de la clase Cnc
- Invoca a la operación d = ObtDirección() de la clase Cnc
- Invoca a la operación ActDireccion(d) de la clase Paralelo, dependiendo del valor de la variable d.
- Invoca a la operación opc = Comando() de la clase Cnc.
- Invoca a la operación [opc = G90]ActAbsoluto(true) de la clase Cnc.
- Invoca a la operación [opc = G91]ActAbsoluto(false) de la clase Cnc.
- Invoca a la operación ActRefParalelo(Rp).

- **Clase Paralelo**

  - **Operación Create():** Crea un objeto P de la clase Paralelo.
  - **Operación ActConectado(True):** P.conectado := true.
  - **Operación ActDireccion(d):** P.direccion := d.
  - **Operación Rp = BuscarRef():** Busca la referencia del objeto P.
  - **Operación ActConectado(False):** P.conectado := false.
  - **Operación Act_pulso(p):** P.pulso := p.
  - **Operación Act_velpulso(vp):** P.velpulso := vp.
  - **Operación Act_vel(v):** P.vel := v.
  - **Operación Act_vel(530):** P.vel := 530.
  - **Operación Act_vel(1060):** P.vel := 1060.
  - **Operación Act_vel(2650):** P.vel := 2650.
  - **Operación Act_vel(5300):** P.vel := 5300.
  - **Operación Act_vel(10600):** P.vel := 10600.
  - **Operación Act_vel(26500):** P.vel := 26500.
  - **Operación ActDireccion(0):** P.direccion := 0
  - **Operación ActDireccion(1):** P.direccion := 1

- **Operación ActDireccion(2):** P.direccion := 2
- **Operación ActDireccion(3):** P.direccion := 3
- **Operación ActDireccion(4):** P.direccion := 4
- **Operación ActDireccion(5):** P.direccion := 5
- **Operación ActDireccion(6):** P.direccion := 6
- **Operación ActDireccion(7):** P.direccion := 7
- **Operación ActDireccion(8):** P.direccion := 8
- **Operación ActDireccion(9):** P.direccion := 9

▪ **Clase Fichero**

- **Operación Create(filName, file):** Crea un objeto F de la clase Fichero, y asigna:
  - F.fileName := fileName.
  - F.file := file.
- **Operación Rf = BuscarRef(file):** Busca la referencia del objeto F con F.file = file.
- **Operación Rf = BuscarRef():** Busca la referencia del objeto F.
- **Operación ActualizarFichero(s):**
  - invoca a la operación FinLista = finLista() de la clase Fichero.
  - Invoca a la operación cadComando = Comando(s) de la clase Fichero.
  - Invoca a la operación lista = Añadir(CadComando) de la clase Fichero.
  - Invoca a la operación Acr_c(lista) de la clase Fichero.
- **Operación FinLista = finLista():** devuelve true o false dependiendo de si ha llegado al fin de la lista o no.
- **Operación cadComando = Comando(s):** Devuelve el comando actual.
- **Operación lista = Añadir(CadComando):** Se añade el comando actual a la lista.

- **Operación Acr_c(lista):** F.v := lista

▪ **Clase Grafico**

- **Operación RepresentarFichero():**
  - invoca a la operación FinLista = finLista() de la clase Grafico.
  - invoca a la operación x_ = ObtenerX() de la clase Gráfico.
  - invoca a la operación y_ = ObtenerY() de la clase Gráfico.
  - invoca a la operación z_ = ObtenerZ() de la clase Gráfico.
  - invoca a la operación ActX(x_) de la clase Gráfico.
  - invoca a la operación ActY(y_) de la clase Gráfico.
  - invoca a la operación ActZ(z_) de la clase Gráfico.
  - invoca a la operación finLinea = finaldelinea() de la clase Grafico.
  - Invoca a la operación [finLinea = true]Act_fin(true) de la clase Grafico.
  - Invoca a la operación [finLinea = false]Act_fin(false) de la clase Grafico.
  - Invoca a la operación opc = Comando() de la clase Grafico
  - Invoca a la operación [opc = G90]ActAbsoluto(true) de la clase Grafico.
  - Invoca a la operación [opc = G91]ActAbsoluto(false) de la clase Grafico.
- **Operación FinLista = finLista():** devuelve true o false dependiendo de si ha llegado al fin de la lista o no.
- **Operación x_ = ObtenerX():** Obtiene la nueva x_.
- **Operación y_ = ObtenerY():** Obtiene la nueva y_.
- **Operación z_ = ObtenerZ():** Obtiene la nueva z_.
- **Operación ActX(x_):** G.x_ := x_.
- **Operación ActY(y_):** G.y_ := y_.
- **Operación ActZ(z_):** G.z_ := z_.

- **Operación finLinea = finaldelinea():** devuelve true o false dependiendo de si ha llegado al fin de la linea o no.
- **Operación [finLinea = true]Act_fin(true):** G.fin_ := true.
- **Operación [finLinea = false]Act_fin(false):** G.fin_ := false.
- **Operación opc = Comando():** Obtiene el comando actual.
- **Operación [opc = G90]ActAbsoluto(true):** G.absoluto := true.
- **Operación [opc = G91]ActAbsoluto(false):** G.absoluto := false.

Capítulo 6

Implementación

En este capítulo hablaremos de todo lo relacionado con la implementación de la aplicación, para ello explicaremos las partes más significativas de la implementación que se ha llevado a cabo para el correcto funcionamiento de la aplicación.

Todo el código fuente del proyecto se encuentra en el CD adjuntado a la memoria.

Para la implementación de la aplicación se ha utilizado de manera integra la programación orientada a objetos y el lenguaje de programación c++, la IDE utilizada para la realización de la implementación así como de la interfaz gráfica ha sido Qt Creator, las librerías gráficas y no gráficas de la aplicación han sido las librerías que nos proporciona Qt, la librería utilizada para la representación gráfica de los ficheros con instrucciones en Codigos Gs ha sido OpenGl, y la librería utilizada para la comunicación vía paralela con la máquina de control numérico fue la librería "inpout32.dll".

6.1. ¿Por qué C++?

C tiene ciertas características sobre otros lenguajes de programación. Los más notables son:

- **Programación orientada a objetos:** La posibilidad de orientar la programación a objetos permite al programador diseñar aplicaciones desde el punto de vista más como una comunicación entre objetos en lugar de en una secuencia estructurada de código. Además, permite una mayor reutilización de código en una forma más lógica y productiva.
- **Portabilidad:** Prácticamente se puede compilar el código C++ mismo en casi cualquier tipo de ordenador y sistema operativo sin realizar ningún

cambio. C++ es el más utilizado y el lenguaje de programación portado en el mundo.

- **Brevedad:** El código escrito en C++ es muy corto en comparación con otros idiomas, ya que el uso de caracteres especiales se prefiere a las palabras clave, ahorrar un poco de esfuerzo para el programador.
- **Programación modular:** El cuerpo de una aplicación en C++ puede estar compuesta de varios archivos de código fuente que se compilan por separado y luego unidas entre sí. Ahorro de tiempo ya que no es necesario volver a compilar la aplicación completa al hacer un solo cambio, pero sólo el archivo que lo contiene. Además, esta característica permite vincular código C con el código producido en otros idiomas, como ensamblador o C.
- **Compatibilidad con C:** C++ es compatible con el lenguaje C. Cualquier código escrito en C pueden ser incluidos en un programa de C sin hacer ningún cambio.
- **Velocidad:** El código resultante de una compilación de C es muy eficiente, por efecto de su dualidad como lenguaje de alto nivel y lenguaje de bajo nivel y el tamaño reducido del propio lenguaje.

6.2. ¿Por qué Programación Orientada a Objetos?

La idea fundamental de la programación orientada a objetos es organizar el sistema en torno a los objetos que intervienen en él, a diferencia de la programación estructurada que se organiza en torno a procesos y datos.

Algunas de las razones por las que es tan atractiva la orientación a objetos son:

- La relativa cercanía de sus conceptos a las entidades que aparecen en el mundo real.
- La simplicidad del modelo, que emplea los mismos elementos fundamentales para expresar de manera uniforme el análisis, el diseño y la implementación de un sistema.
- La gran capacidad de adaptación e integración de sus modelos, que facilita la realización de modificaciones, incluso durante el proceso de desarrollo, y el mantenimiento.
- La posibilidad de aumentar las oportunidades de reutilización de componentes de software en proyectos distintos.

Los conceptos mas significativos de la programación orientada a objetos son:

- **Abstracción:** denota las características esenciales de un objeto, donde se capturan sus comportamientos.Cada objeto en el sistema sirve como modelo de un "agente" abstracto que puede realizar trabajo, informar y cambiar su estado, y "comunicarse" con otros objetos en el sistema sin revelar cómo se implementan estas características.La abstracción es clave en el proceso de análisis y diseño orientado a objetos, ya que mediante ella podemos llegar a armar un conjunto de clases que permitan modelar la realidad o el problema que se quiere atacar.

- **Encapsulamiento:** Significa reunir a todos los elementos que pueden considerarse pertenecientes a una misma entidad, al mismo nivel de abstracción. Esto permite aumentar la cohesión de los componentes del sistema.

- **Modularidad:** Se denomina Modularidad a la propiedad que permite subdividir una aplicación en partes más pequeñas (llamadas módulos), cada una de las cuales debe ser tan independiente como sea posible de la aplicación en sí y de las restantes partes. Estos módulos que se puedan compilar por separado, pero que tienen conexiones con otros módulos.

- **Principio de ocultación:** Cada objeto está aislado del exterior, es un módulo natural, y cada tipo de objeto expone una interfaz a otros objetos que especifica cómo pueden interactuar con los objetos de la clase. El aislamiento protege a las propiedades de un objeto contra su modificación por quien no tenga derecho a acceder a ellas, solamente los propios métodos internos del objeto pueden acceder a su estado.

- **Polimorfismo:** comportamientos diferentes, asociados a objetos distintos, pueden compartir el mismo nombre, al llamarlos por ese nombre se utilizará el comportamiento correspondiente al objeto que se esté usando. Cuando esto ocurre en "tiempo de ejecución", esta última característica se llama asignación tardía o asignación dinámica. Algunos lenguajes proporcionan medios más estáticos (en "tiempo de compilación") de polimorfismo, tales como las plantillas y la sobrecarga de operadores de C++.

- **Herencia:** las clases no están aisladas, sino que se relacionan entre sí, formando una jerarquía de clasificación. Los objetos heredan las propiedades y el comportamiento de todas las clases a las que pertenecen. La herencia organiza y facilita el polimorfismo y el encapsulamiento permitiendo a los objetos ser definidos y creados como tipos especializados de objetos preexistentes.

6.3. ¿Por qué Qt?

Qt es una biblioteca multiplataforma para desarrollar interfaces gráficas de usuario y también para el desarrollo de programas sin interfaz gráfica como herramientas de la consola y servidores.

Qt es utilizada principalmente en Autodesk, KDE, Google Earth, la Agencia Espacial Europea, Skype, Qt Extended, Adobe Photoshop Album, VirtualBox, VLC media player, Samsung, Philips, Panasonic y Opie.

Funciona en todas las principales plataformas, y tiene un amplio apoyo. El API de la biblioteca cuenta con métodos para acceder a bases de datos mediante SQL, así como uso de XML, gestión de hilos, soporte de red, una API multiplataforma unificada para la manipulación de archivos y una multitud de otros para el manejo de ficheros, además de estructuras de datos tradicionales.

Qt está distribuida bajo los términos de GNU Lesser General Public License (y otras), además Qt es software libre y de código abierto.

6.4. ¿Por qué OpenGL?

OpenGL (Open Graphics Library) es una especificación estándar que define una API multilenguaje y multiplataforma para escribir aplicaciones que produzcan gráficos 2D y 3D. La interfaz consiste en más de 250 funciones diferentes que pueden usarse para dibujar escenas tridimensionales complejas a partir de primitivas geométricas simples, tales como puntos, líneas y triángulos.

OpenGL tiene dos propósitos esenciales:

- Ocultar la complejidad de la interfaz con las diferentes tarjetas gráficas, presentando al programador una API única y uniforme.
- Ocultar las diferentes capacidades de las diversas plataformas hardware, requiriendo que todas las implementaciones soporten la funcionalidad completa de OpenGL.

El funcionamiento básico de OpenGL consiste en aceptar primitivas tales como puntos, líneas y polígonos, y convertirlas en pixels. Este proceso es realizado por una pipeline gráfica conocida como Máquina de estados de OpenGL. La mayor parte de los comandos de OpenGL bien emiten primitivas a la pipeline gráfica o bien configuran cómo la pipeline procesa dichas primitivas.

OpenGL es una API basada en procedimientos de bajo nivel que requiere que el programador dicte los pasos exactos necesarios para renderizar una escena. Esto contrasta con las APIs descriptivas, donde un programador sólo debe describir la escena y puede dejar que la biblioteca controle los detalles para representarla. El diseño de bajo nivel de OpenGL requiere que los programadores conozcan en profundidad la pipeline gráfica, a cambio de darles libertad para implementar algoritmos gráficos novedosos.

6.5. ¿Por qué la librería inpout32?

Escribir programas con el fin de comunicarse con el puerto paralelo era bastante sencillo en la época de DOS y también en Win95/98. Podríamos utilizar Inporb y funciones outportb o _inp () o _Outp en nuestro programa sin ningún problema si se está ejecutando el programa en DOS o Windows 95/98. Pero con la llegada de los sistemas operativos NT, Win2000, WinXP ... , toda esta simplicidad se va.

Cuando estamos tratando de ejecutar un programa que escribiendo funciones convencionales como Inporb, outportb, _inp () o _Outp en un sistema WINXP o WIN2000, se mostrará un mensaje de error "La excepción de instrucción privilegiada ".

Siendo un sistema operativo muy seguro, Windows XP asigna algunos privilegios y restricciones a los diferentes tipos de programas que se ejecutan en el. Clasifica todos los programas en dos categorías, de modo de usuario y el modo de núcleo, es decir, se ejecuta en modos ring3 y ring0.

Los programas de modo de usuario se ejecutan en modo de ring3 y los programas de modo de núcleo se ejecutan en modo de ring0. Los programas por lo general, caen en la categoría de modo de usuario. Los programas de modo de usuario se limitan a utilizar ciertas instrucciones, como IN OUT etc ...Cada vez que el sistema operativo encuentra un programa en modo de usuario está tratando de ejecutar tales instrucciones, el sistema operativo detiene la ejecución de esos programas y se mostrará un mensaje de error.

La solución a este problema fue utilizar Inpout32.dll en sistemas operativos como Win98/NT/2000/XP. Esta dll tienen las siguientes características:

1. Funciona con todas las versiones de Windows (Win 98, NT, 200 y XP).
2. El uso de un controlador en modo de núcleo incrustado en el archivo DLL.
3. No es necesario ningún software especial o instalar drivers.
4. El Controlador se instala automáticamente y se configura automáticamente cuando el archivo DLL se carga.
5. No es necesaria una API especial sólo dos funciones Inp32 y Out32 .
6. Puede ser fácilmente utilizada con VC y VB.

6.6. Implementaciones reseñáble

En esta sección podemos ver los fragmentos de código más relevantes del proyecto.

6.6.1. Mover motor en tiempo real

```
void CNC::MoverUp()
{
    if((ui->y_abs->value() + incremento) < tamBase)
    {
        if(paralelo->conectado_){
            paralelo->direccion = 2;
            paralelo->start(QThread::TimeCriticalPriority);
        }

        yr = ui->y_rel->value() + incremento;
        ui->y_rel->display(yr);
        ya = ui->y_abs->value() + incremento;
        ui->y_abs->display(ya);
    }
}
```

En este caso podemos ver como movemos el motor del eje Y, positivamente.

6.6.2. Procesar fichero

```
void Fichero::Procesar(QString s)
{
    QStringList list;
    QStringList lista;

    if(s.at(0) == QChar('N'))
    {
            ...

        // Si la instrucción contiene G90
        if(s.contains("G90", Qt::CaseInsensitive))
        {
            lista.append("G90");     // Añadimos 'G90' a la lista
            v.push_back(lista);  // Añadimos la lista al vector
        }
            ...

        if(s.contains("G00X",Qt::CaseInsensitive)
            ||s.contains("G00Y",Qt::CaseInsensitive
            || s.contains("G00Z", Qt::CaseInsensitive))
        {
            lista.push_front("G00");
            v.push_back(lista);
```

```
        }
            ...
    }
}
```

En este caso podemos ver cómo procesamos una instrucción G90 y una instrucción G00.

6.6.3. Representar fichero graficamente

```
void Grafico::draw()
{
    ...

    if(((*i).first()).contains("G90",Qt::CaseInsensitive)){}     //ABSOLUTO
    if(((*i).first()).contains("G91",Qt::CaseInsensitive)){}     //INCREMENTAL

    if((*i)[0].contains("G1", Qt::CaseInsensitive))
    {
        double x;    // Definimos 3 variables para las 3 coordenadas
        double y;
        double z;

        x = x_;    // Asignamos la posición actual a las 3 coordendas
        y = y_;
        z = z_;

        if((*i)[1] != "$")  // Comprobamos si x está definido
        {
        if(!absoluto) x = x + (*i)[1].toDouble(); // Si es incremental sumamos
        else x = (*i)[1].toDouble();              // Si es absoluto asignamos
        }

        if((*i)[2] != "$")
        {
            if(!absoluto)y = y + (*i)[2].toDouble();
            else y = (*i)[2].toDouble();
        }

        if((*i)[3] != "$")
        {
            if(!absoluto)z = z + (*i)[3].toDouble();
            else z = (*i)[3].toDouble();
        }
```

```
            glBegin(GL_LINES);                  // Pintamos una linea mediante OpenGL
                qglColor(Qt::red);              // Elegimos el color de la linea
                glVertex3f(x_,y_,z_);           // Coordenadas de origen
                glVertex3f(x,y,z);              // Coordenadas de destino
            glEnd();

            ...
        }
        ...
}
```

En este fragmento de código obsevamos cómo pintamos mediante OpenGL una la linea generada por la instrucción G1.

6.6.4. Ejecución

```
void CNC::Run()
{
    if(abierto == true)
    {
        QVector<QStringList> *R = new QVector<QStringList
            >(f_->Codigos());

        double x = ui->x_abs->value();
        double y = ui->y_abs->value();
        double z = ui->z_abs->value();

        if(paralelo->conectado_)
        {
            ...

            if(((*i).first()).contains("G00", Qt::
                CaseInsensitive))
            {
                paralelo->vel = 530;          //-> 1 mm
                double x2 = 0;
                double y2 = 0;
                double z2 = 0;

                double xi = 0;
                double yi = 0;
                double zi = 0;

                if((*i)[1] != "$")
                {
```

```
            if(!absoluto) x2 = x + (*i)[1].
                toDouble();
            else x2 = (*i)[1].toDouble();
        }else x2 = x;

        ...

        xa = x;
        ya = y;
        za = z;

        //distancia individual
        if(x2 != x) xi = qAbs((x2-x));
            else xi = 0;
        if(y2 != y) yi = qAbs((y2-y));
            else yi = 0;
        if(z2 != z) zi = qAbs((z2-z));
            else zi = 0;

        //distancia euclidea
        int euclidea = qSqrt(xi*xi + yi*yi);

        double dx = xi/euclidea;
        double dy = yi/euclidea;

        //distancia a recorrer en cada iteracion
        int velx = dx*530;
        int vely = dy*530;
        int velz = zi*530;

        if(zi > 0)
        {
            paralelo->vel = velz;
            if(z2 >= za) paralelo->direccion = 4;
            else paralelo->direccion = 5;

            paralelo->start(QThread::
                TimeCriticalPriority);
        }

    // si es vertical u horizontal o digonal 45º,
        se recorre de una vez
    if(xi == 0 || yi == 0 || xi == yi)
    {
        // mover en diagonal
        if(xi == yi)
```

```
        {
            velx = xi*530;
            paralelo->vel = velx;

        if(x2 >= xa && y2 >= ya)
            {paralelo->direccion = 6;}
        else if(x2 >= xa && y2 < ya)
            {paralelo->direccion = 7;}
        else if(x2 < xa && y2 >= ya)
            {paralelo->direccion = 8;}
        else if(x2 < xa && y2 < ya)
            {paralelo->direccion = 9;}

        paralelo->start(QThread::
            TimeCriticalPriority);

        }else{
            //MOVER X ...
            //MOVER Y ...
        }
    }else{

        // creamos la linea por pasos
    while(euclidea > 0)
    {
        //MOVER X
        if(dx > 0)
        {
            paralelo->vel = velx;
            if(x2 >= xa) paralelo->direccion = 0;
            else paralelo->direccion = 1;

            paralelo->start(QThread::
                TimeCriticalPriority);
                                                    }
        //MOVER Y    ...

        euclidea--;
        }
    }

    // actualizamos valores
    xa = x2;
    xr = ui->x_rel->value() + (x2-x);
    ya = y2;
    yr = ui->y_rel->value() + (y2-y);
```

```
            za = z2;
            zr = ui->z_rel->value() + (z2-z);

            //mostramos desplazamiento absoluto
            ui->x_abs->display(xa);
            ui->y_abs->display(ya);
            ui->z_abs->display(za);

            //mostramos desplazamiento relativo
            ui->x_rel->display(xr);
            ui->y_rel->display(yr);
            ui->z_rel->display(zr);

            x = ui->x_abs->value();
            y = ui->y_abs->value();
            z = ui->z_abs->value();
        }// cerrar G00
            ...
    }...
}...}
```

Aquí podemos ver un bloque de código correspondiente a la ejecución de una instrucción G00.

6.6.5. Carga de librería inpout32.dll

```
Paralelo::Paralelo()
{
    HINSTANCE hLib;

    hLib = LoadLibrary(TEXT("inpout32.dll"));      // Abrimos librería

    if (hLib == NULL)
        qDebug("ERROR: LoadLibrary Failed.\n");

    inp32 = (inpfuncPtr) GetProcAddress(hLib, "Inp32");
    if (inp32 == NULL) qDebug("GetProcAddress for Inp32 Failed.\n");

    oup32 = (oupfuncPtr) GetProcAddress(hLib, "Out32");
    if (oup32 == NULL) qDebug("GetProcAddress for Oup32 Failed.\n");

    oup32(0x37A, 0x0C);
    conectado_ = false;
}
```

6.6.6. Hilo de ejecución

```
void Paralelo::run()
{
    switch(direccion)
    {
        case 0: // X+
            for(int i = 0; i < vel; i++){   // un pulso
                oup32(0x378, 0x03); // flanco de subida precisión en us
                    sleep(pulso);

                oup32(0x378, 0x02); // flanco de bajada precisión en us
                    sleep(velpulso);
            }
        break;

        case 1: // X-
            for(int i = 0; i < vel; i++){
                oup32(0x378, 0x01);
                    sleep(pulso);
                oup32(0x378, 0x00);
                    sleep(velpulso);
            }
        break;
                    ...
    }
}
```

En este fragmento de código vemos cómo movemos los motores enviándole pulsos.

Capítulo 7

Pruebas

La fase de pruebas es una de las más costosas del ciclo de vida software. En sentido estricto, deben realizarse pruebas de todos los artefactos generados durante la construcción de un producto, lo que incluye especificaciones de requisitos, casos de uso, diagramas de diversos tipos y, por supuesto, el código fuente y el resto de productos que forman parte de la aplicación.

Las pruebas realizadas al sistema fueron pruebas de caja blanca y pruebas de caja negra, estas pruebas se hicieron en 2 fases temporales, durante el desarrollo de la aplicación y una vez que se finalizó el producto.

7.1. Pruebas de caja blanca

Las pruebas de caja blanca realizan un seguimiento del código fuente según va ejecutando los casos de prueba, de manera que se determinan de manera concreta las instrucciones, bloques, etc. en los que existen errores. Cuando se pasan casos de prueba al programa que se está probando, es conveniente conocer qué porcentaje del programa se ha ejecutado, de manera que estemos próximos a asegurar que todo él es correcto. En nuestro caso llevamos a cabo varias formas de medir la cobertura lograda en el programa por los casos de prueba.

- **Cobertura de sentencia:** Cada instrucción del programa debe ejecutarse al menos una vez.
- **Cobertura de decisión:** Comprueba el número de decisiones ejecutadas, considerando que se ha ejecutado una decisión cuando se han recorrido todas sus posible ramas (la que la hace true y la que la hace false, pero también todas las posibles ramas de un switch).
- **Cobertura de condiciones:** Comprueba el número de condiciones múltiples ejecutadas, considerando que se ha ejecutado una condición múltiple

cuando se han ejecutado todas sus correspondientes ramas con todas las posibles variantes de la instrucción condicional.

- **Cobertura de condición/decisión:**Comprueba el número de condiciones y decisiones que se han ejecutado.

7.2. Pruebas de caja negra

Esta prueba consiste en tomar el software o una parte de este como un bloque, al cual no se tiene acceso de una manera interna.

Con respecto al software realizamos dos tipos de pruebas:

- **Prueba de equivalencia:** Según las restricciones de formato o contenido de los parámetros de entrada, identicaremos clases en las que se englobarán dichos parámetros. Una vez establecidas dichas clases se comprobará si todos los elementos de dicha clase son tratados por igual. Probaremos las marcas de dichas clases para comprobar que todos los elementos de una clase son tratados de igual forma.
- **Prueba de valores límites:** En estas pruebas se van dando entrada a valores límites de cada campo, como pueden ser cadenas vacías, muy largas, el número cero si se trata de enteros, o valores nulos.

Con respecto al hardware se realizaron numerosas pruebas la mayor parte de las pruebas fueron, pruebas de comunicación por puerto serie y pruebas de comunicación por puerto paralelo.

En las pruebas de comnunicación por puerto paralelo, se llevaron a cabo por medio de un analizador lógico donde observábamos con más precisión y detenimiento tanto los tiempos como la duración de cada pulso de salida.

Esta fue una de las pruebas iniciales, lejana a lo que se esperaba, 3,8 us de pulso(A->B), y ~135 us de retardo.(B->C)

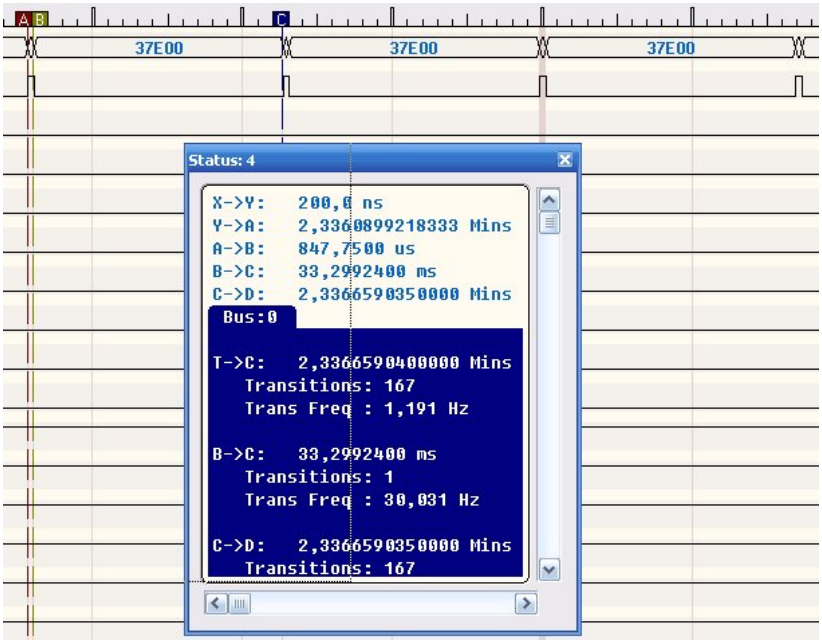

Figura 7.1: Primera Prueba

Más adelante se realizó una segunda prueba con dando estos resultados más cercanos:

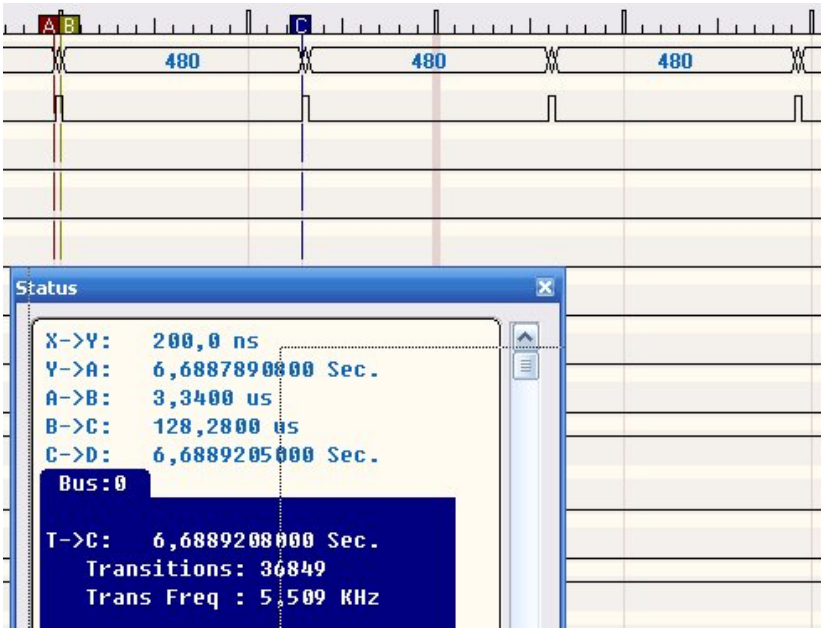

Figura 7.2: Segunda Prueba

Una tercera prueba general de todas las posibles direcciones de la máquina dio los siguientes resultados resultados:

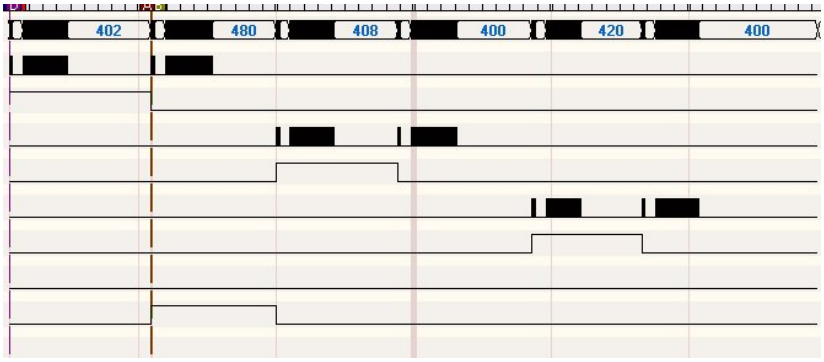

Figura 7.3: Prueba General

Aquí podemos ver una prueba que se realizó en el arranque de la máquina con nuestra aplicación, más adelante se movió el eje X en sentido negativo.

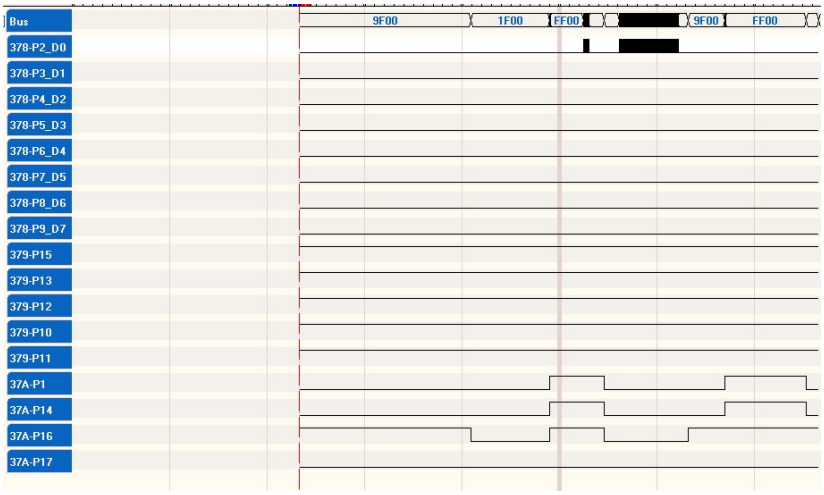

Figura 7.4: Prueba de arranque

Para controlar mejor los tiempos se utilizó la función usleep que controlaba los microsegundos de retardo, aunque dio lugar a nefastos resultados como vemos en la siguiente prueba:

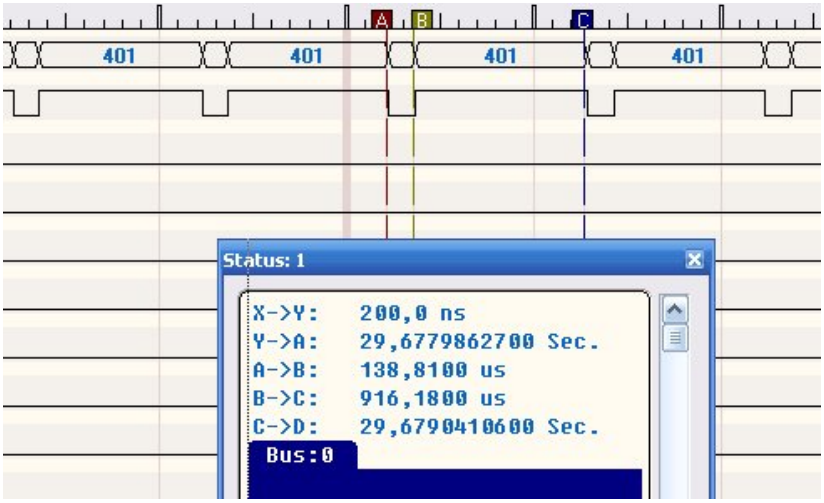

Figura 7.5: Prueba uSleep

La siguiente prueba representa una prueba final en la que vemos que se dieron los resultados esperados por la aplicación.

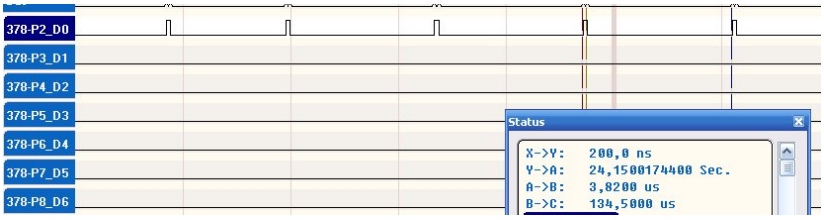

Figura 7.6: Prueba final

Además de pruebas de caja blanca y pruebas de caja negra, se realizaron pruebas de unidad pruebas de integración incremental y pruebas del sistema por este orden.

7.3. Pruebas de unidad

Se trata de las pruebas formales que permiten declarar que un módulo está listo y terminado.

Hablamos de una unidad de prueba para referirnos a uno o más módulos que cumplen las siguientes condiciones [IEEE, 1986a]:

- Todos son del mismo programa
- Al menos uno de ellos no ha sido probado
- El conjunto de módulos es el objeto de un proceso de prueba

La prueba de unidad puede abarcar desde un módulo hasta un grupo de módulos.

Estas pruebas suelen realizarlas el propio personal de desarrollo, pero evitando que sea el propio programador del módulo.

7.4. Pruebas de integración

Implican una progresión ordenada de pruebas que van desde los componentes o módulos y que culminan en el sistema completo.
El orden de integración elegido afecta a diversos factores, como lo siguientes:

- La forma de preparar casos
- Las herramientas necesarias

- El orden de codificar y probar los módulos
- El coste de la depuración
- El coste de preparación de casos

En Integración incremental que es la que hemos utilizado se combina el siguiente módulo que se debe probar con el conjunto de módulos que ya han sido probados.

7.5. Pruebas del sistema

Es el proceso de prueba de un sistema integrado de hardware y software para comprobar lo siguiente:

- Cumplimiento de todos los requisitos funcionales, considerando el producto software final al completo en un entorno de sistema.
- El funcionamiento y rendimiento en las interfaces hardware, software, de usuario y de operador.
- Adecuación de la documentación de usuario.
- Ejecución y rendimiento en condiciones límite y de sobrecarga.

Capítulo 8

Conclusiones

8.1. Conclusiones del proyecto realizado

La realización de este proyecto de fin de carrera me ha servido de mucho, ya que he conseguido aplicar todos los conocimientos obtenidos a lo largo de los años cursados de ingeniería técnica, y además he tenido la oportunidad de aprender nuevos conceptos, diferentes a los adquiridos a lo largo de la carrera.

En el transcurro de la realización de este proyecto he aprendido muchos conceptos relacionados con las máquinas de control numérico, su funcionamiento, características, estructura y partes que la conforman, además de cómo manejar este tipo de máquinas.

Además de aprender todo lo relacionado con las máquinas de control numérico, he aprendido a realizar programas mediante comandos G, y saber interpretarlos correctamente.

Otros grandes conocimientos que he adquirido con la realización del proyecto, ha sido el desarrollo de comunicaciones serie por puerto usb, y comunicaciones mediante puerto paralelo, que ha servido para comunicarme con la máquina de control numérico.También he aprendido a utilizar algo tan útil como es un analizador lógico.

Con respecto a la implementación, he aplicado mis conocimientos del lenguaje de programación C++, utilizándolos junto con las librerías Qt en el entorno de desarrollo integrado Qt Creator, que me ha servido para la realización integra del proyecto.

Otro aspecto que he podido aplicar en la realización de este proyecto ha sido, todos los aspectos relacionados con la ingeniería del software, en este caso

orientada a objetos, realizando un análisis y diseño exhaustivo de la aplicación final.

Además de todo lo comentado anteriormente he puesto en práctica mis conocimientos en ingles, ya que la mayor parte de la documentación, por ejemplo de Qt, estaba en el idioma anglosajón.

Por último decir también que, mediante la realización de esta memoria del proyecto he afianzado e incrementado mis conocimientos sobre la composición de textos por medio de LATEX.

8.2. Futuras mejoras del proyecto

Para futuras versiones del producto hay puestas varias ideas que harían de el, un mejor producto. Estas son algunas de ellas:

- Poder modificar el fichero dentro de la aplicación.
- Poder desarrollar nuevos ficheros en la aplicación.
- Aceptar mas variedad de ficheros de códigos G.
- Ampliar el numero de códigos G y M procesados por la aplicación.
- Poder realizar un editor de figuras 3d y mediante él, producir un fichero posteriormente ejecutable por la aplicación.
- Poder modificar las características de la herramienta utilizada.
- Poder modificar el sistema de velocidad y aceleración de pulsos.
- Desarrollar versiones para diversos sistemas operativos, principalmente Linux.

Capítulo 9

Manual de instalación

A continuación explicaremos los pasos a tomar por parte del usuario para la correcta instalación de la aplicación en su PC.

9.1. Requisitos previos

Para disfrutar de todas las funcionalidades para las que ha sido desarrollado el producto el cliente debe de poseer:

- Una máquina de control numérico, ya que la aplicación está diseñada con el fin de interactuar con tal herramienta.
- Poseer un ordenador con puerto paralelo, ya que en el caso de carecer de ella la comunicación con la máquina seria inviable.
- Tener instalado un sistema operativo windows, ya que la aplicación por el momento solo funciona bajo tal sistema operativo.

9.2. Instalación del producto

Estos son los pasos que debe tomar el usuario para una correcta instalación del producto.

Paso 1: Ejecutar el instalador: *Setup.exe*

Figura 9.1: Instalación: Seleccionar idioma

Paso 2: Seleccionar el idioma de la instalación.

Paso 3: Seguir las indicaciones del instalador.

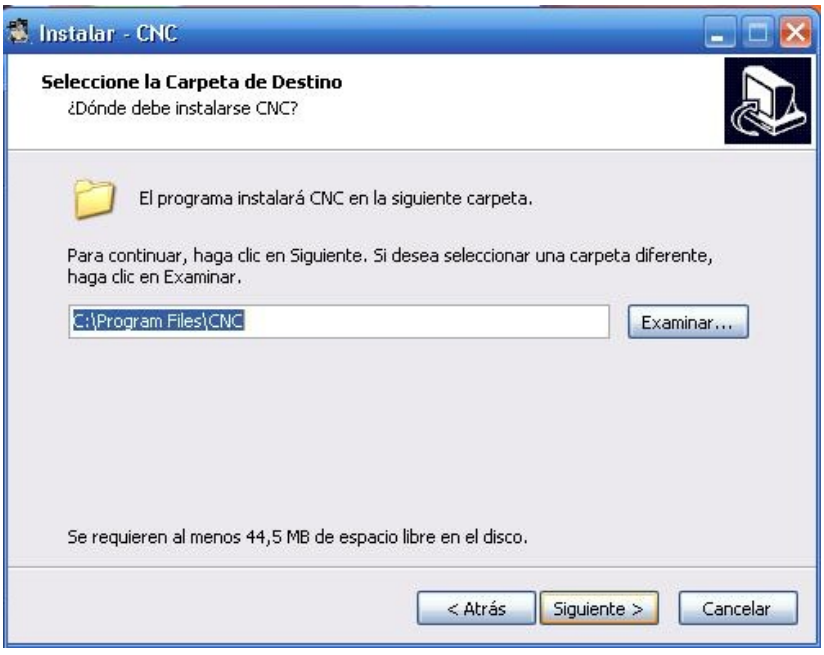

Figura 9.2: Instalación: Carpeta de destino

Paso 4: Seleccionar carpeta de destino de la instalación de la aplicación

Figura 9.3: Instalación: Carpeta del menú de inicio

Paso 5: Seleccionar carpeta del menú de inicio

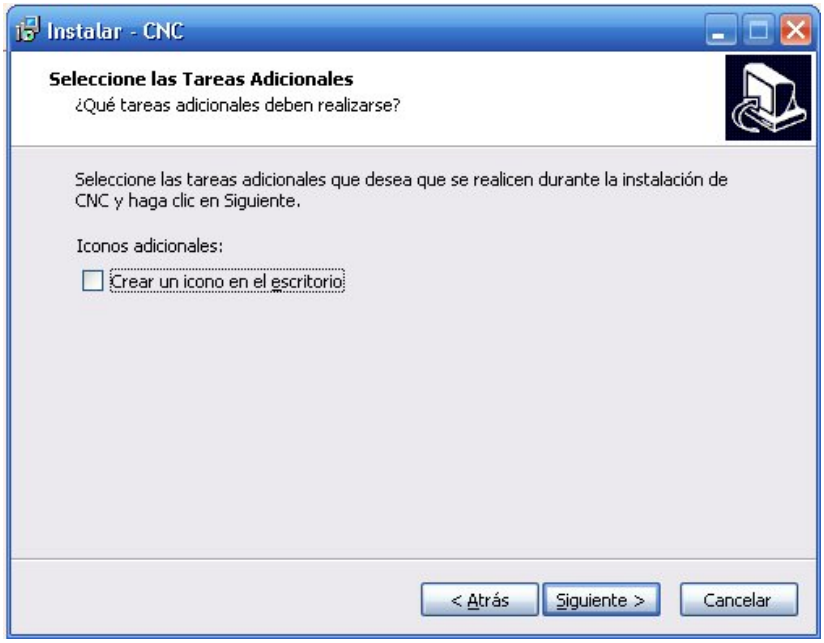

Figura 9.4: Instalación: Tareas adicionales

Paso 6: Indicamos la posibilidad de incluir un icono en el escritorio

Figura 9.5: Instalación: Instalación completada

Paso 7: Seguimos los pasos hasta finalizar la instalación.

Capítulo 10

Manual de usuario

En este capítulo se describirá cada una de las funcionalidades del producto así como su utilización adecuada con respecto a las necesidades del usuario.

10.1. Visión general

En esta sección vemos el aspecto general de la aplicación en el momento de su ejecución.

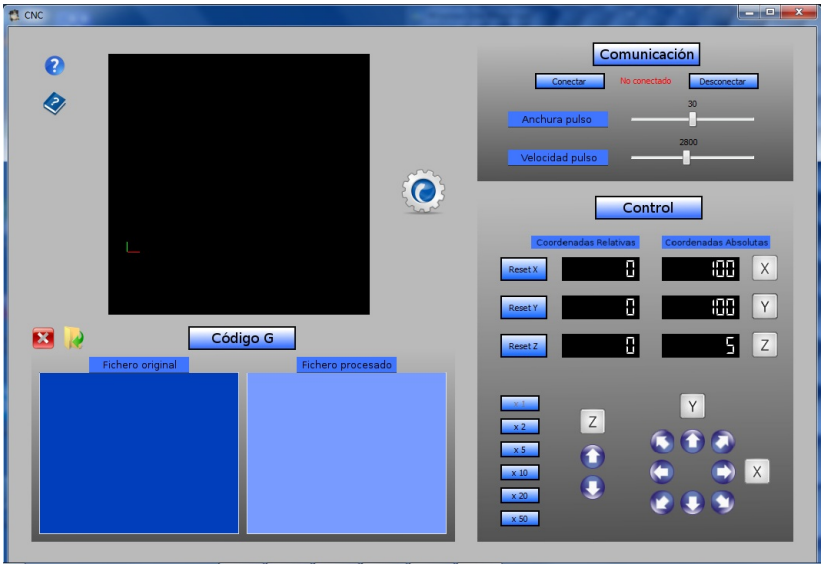

Figura 10.1: Visión general de la aplicación

Por medio de está imagen podemos ver que la aplicación se divide principalmente en 4 partes:

- Una primera parte de comunicación con la máquina.
- Sección de control de la máquina.
- Visor del fichero con códigos Gs y su procesamiento.
- Y una ultima parte donde vemos el gráfico producido por el fichero.
- Tenemos también un botón de ejecución y la ayuda de la aplicación

10.2. Establecer comunicación con la máquina CNC

En esta sección vemos como podemos iniciar y terminar la comunicación con la máquina CNC previamente conectada.

En el momento de inicio de la aplicación el sistema está no conectado y tiene la anchura y velocidad de pulso por defecto.

Figura 10.2: Comunicación inicial

Al pulsar el botón “Conectar”, nos conectaríamos a la máquina, siempre y cuando todo esté correcto a nivel hardware y tengamos la máquina conectada a la corriente y al ordenador mediante el puerto paralelo.

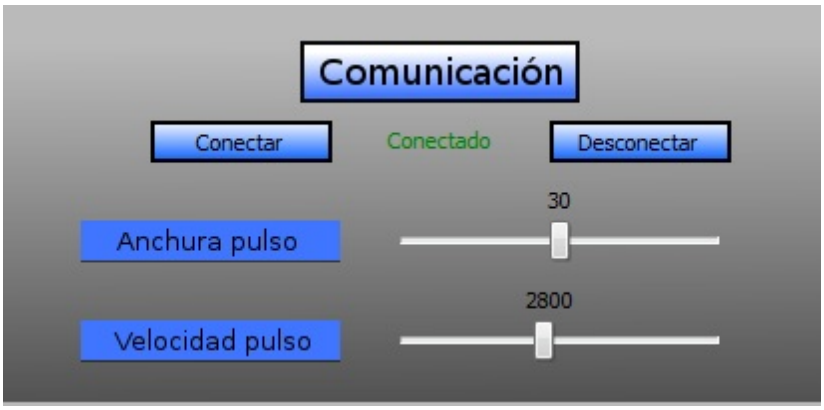

Figura 10.3: Comunicación con éxito

Una vez que haya tenido éxito la vinculación de nuestra aplicación con la máquina cnc, podríamos ejecutar cualquier fichero cnc o interactuar con ella.

10.3. Control de la máquina cnc

En está sección podemos ver de que forma podemos controlar la máquina mediante la aplicación y como se vería reflejado esto en nuestra aplicación.

Abajo podemos ver qué parte de la aplicación se dedica al control de la máquina.

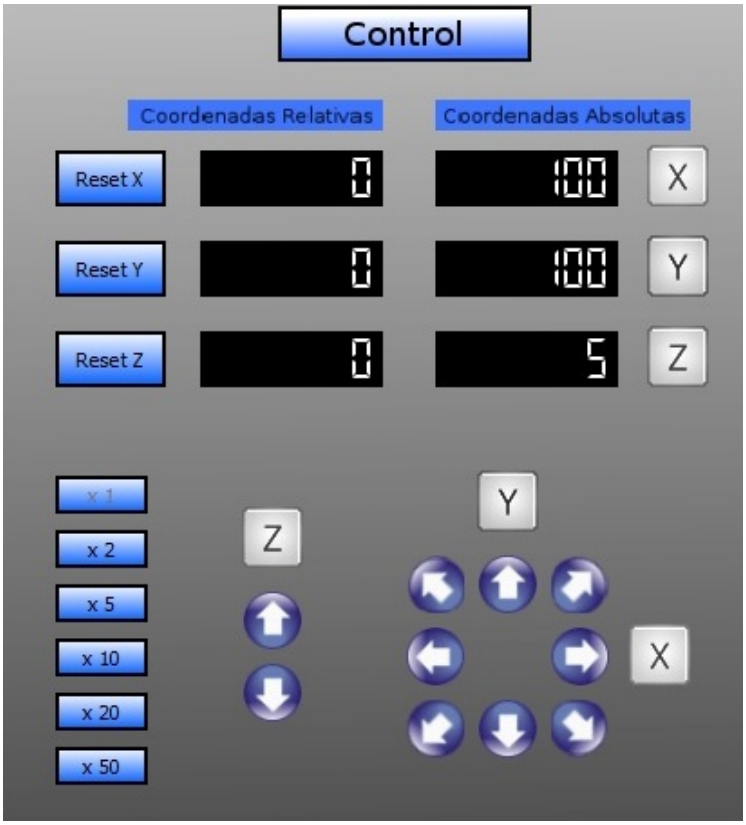

Figura 10.4: Control de la máquina

En la figura 9.4 podemos ver que la parte dedicada al control de la máquina tiene 2 partes bien diferenciadas:

- Una primera parte donde vemos la posición de la máquina en el momento actual
- Una segunda parte donde se controla el movimiento de la máquina.

10.3.1. Visor del posicionamiento de la máquina

Figura 10.5: Posicionamiento de la máquina

En la figura 9.5 podemos ver 3 partes:

- Una primera parte donde vemos las coordenadas absolutas x, y, z.
- Una segunda parte donde podemos observar las coordenadas relativas x, y, z.
- Una tercera parte donde tenemos 3 botones capaces de resetear las coordenadas relativas x, y, z.

10.3.2. Control del movimiento de la máquina

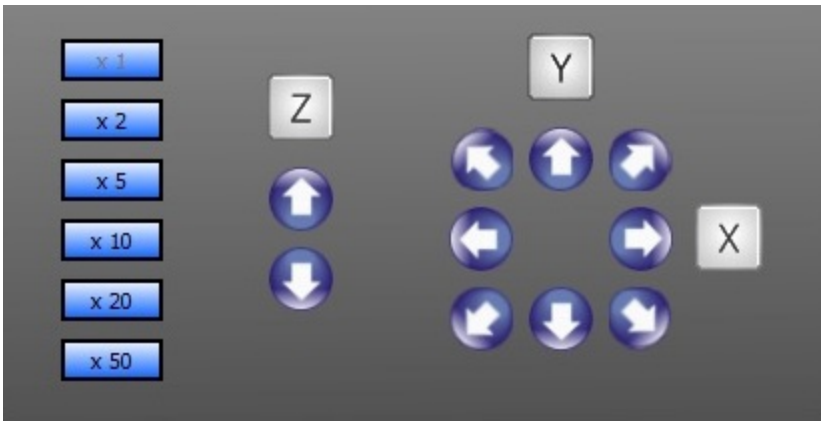

Figura 10.6: Control del movimiento de la máquina

En la figura 9.6 podemos ver que tenemos 2 partes diferenciadas dentro del control del movimiento de la máquina

- Una primera parte donde controlamos la cantidad de pulsos enviados a la máquina, x1, x2, x5 ...
- Una segunda parte donde controlamos los motores de la máquina siendo posible moverlo en todas las direcciones posibles del eje X, Y y Z.

10.3.2.1. Control de la máquina mediante teclado

El movimiento de la máquina además de poder controlarlo mediante la interfaz gráfica, también se puede controlar mediante el teclado

Aquí tenemos una imagen ilustrativa de las teclas útiles del teclado para esta aplicación.

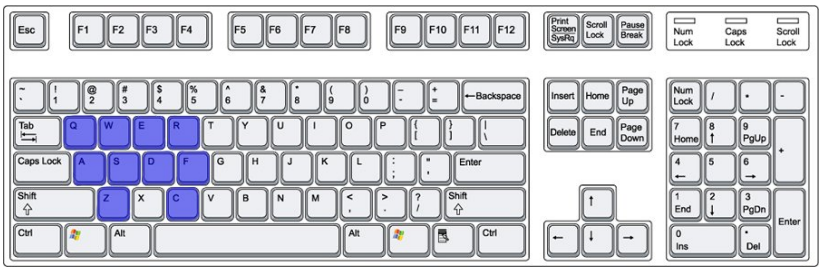

Figura 10.7: Control del movimiento de la máquina

-  Mueve el motor del eje X en sentido positivo
-  Mueve el motor del eje X en sentido negativo
-  Mueve el motor del eje Y en sentido positivo
-  Mueve el motor del eje Y en sentido negativo
-  Mueve el motor del eje Z en sentido positivo
-  Mueve el motor del eje Z en sentido negativo
-  Mueve el motor del eje X en sentido positivo e Y en sentido positivo
-  Mueve el motor del eje X en sentido positivo e Y en sentido negativo

-  Mueve el motor del eje X en sentido negativo e Y en sentido positivo
-  Mueve el motor del eje X en sentido negativo e Y en sentido negativo

10.4. Visor de fichero con códigos G

En esta sección podemos ver como abrimos, procesamos y cerramos un fichero con códigos g.

Abajo podemos ver el aspecto de la parte de la aplicación que se ocupa de esto, al iniciar la aplicación.

Figura 10.8: Visor de fichero código G

Para abrir un fichero con códigos g, pulsamos el botón y a continuación buscamos el fichero que nos interese abrir.

Una vez elegido tendremos algo parecido a esto:

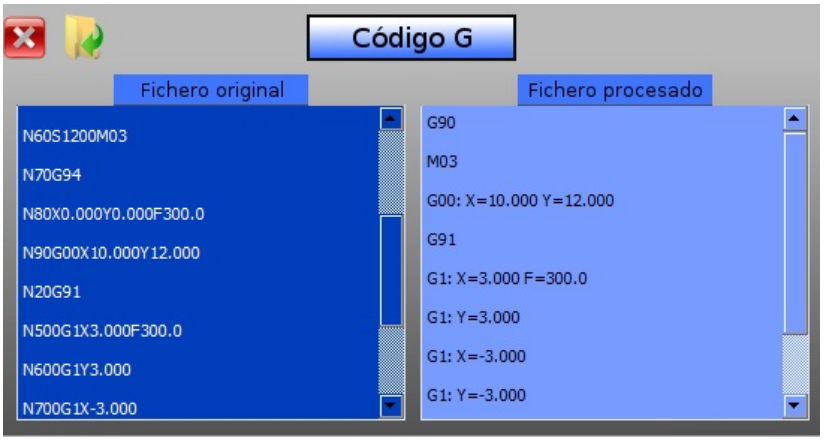

Figura 10.9: Fichero abierto

En la figura 9.8 podemos ver como en el visor de la izquierda aparece el

fichero cnc abierto sin filtrar, y en la parte de la derecha vemos el fichero ya filtrado con las instrucciones que entiende la aplicación. Una vez abierto y filtrado el fichero, podremos ver el gráfico que produce.

Para cerrar el fichero sólo tenemos que pulsar el botón  .

10.5. Visor gráfico

Aquí podemos ver como se representa el fichero con códigos Gs válidos al abrirlo y filtrarlo.

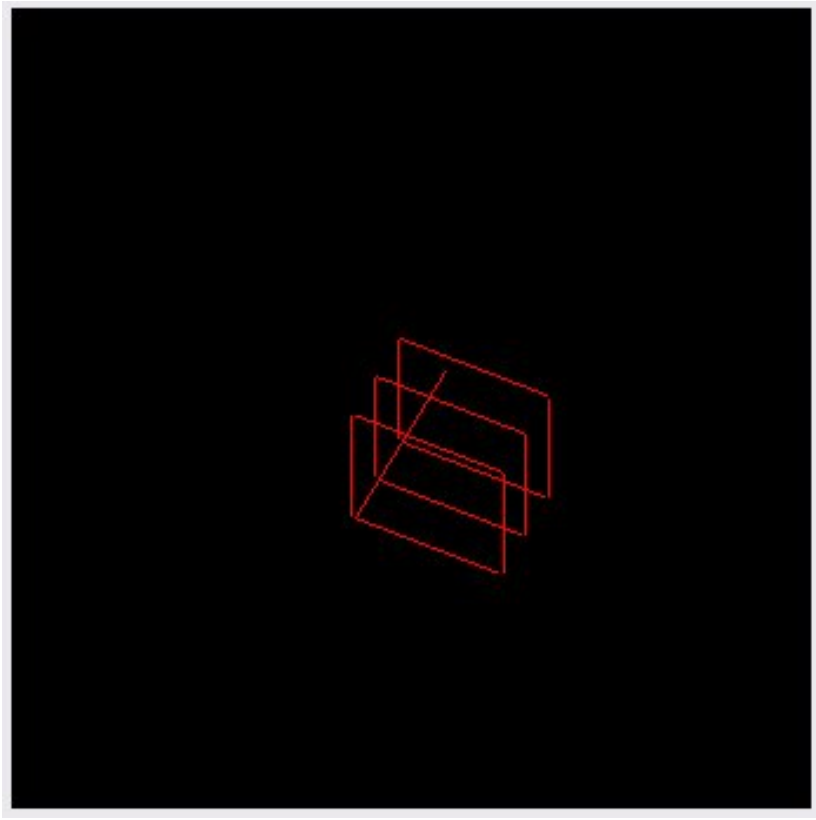

Figura 10.10: Visor gráfico

Por medio del ratón podemos:

- Con el botón izquierdo del ratón, rotamos el dibujo generado por el fichero.
- Con el botón derecho del ratón, trasladamos el dibujo generado por el fichero.
- Con el botón central del ratón, acercamos o alejamos el dibujo generado por el fichero.

10.6. Ejecución del programa

Una vez abierto el fichero y establecida la comunicación con la máquina cnc, podemos ejecutar el programa, haciendo esto trabajar a la máquina de manera automática siguiendo las instrucciones establecidas mediante el fichero cnc abierto con anterioridad.

Para ello sólo tenemos que pulsar el siguiente botón:  .

10.7. Ayuda de la aplicación

Para llegar hasta este manual de ayuda sólo se ha tenido que pulsar el botón  , y para acceder a la información acerca de la aplicación sólo tenemos que pulsar el botón  .

Capítulo 11

Bibliografía y referencias

[Pressman] Ingeniería del software un enfoque práctico. Mc Graw Hill

[Métrica 3] **Administración eletrónica del gobierno de España**

[Craig Larman] UML y Patrones

[Blanchette] C++ GUI programming with Qt 4

[Qt] Manual de referencia de Qt **http://doc.qt.nokia.com/**

[Addison] OpenGL programming guide

[Peter Smid] CNC programming handbook

[Suk] Theory and Design of CNC Systems

[Códigos G] The NIST RS274NGC Standard

Apéndice A

Códigos G y Códigos M

CÓDIGOS G	
CÓDIGO	**FUNCIÓN**
G00	Movimiento rápido
G01	Movimiento de interpolación lineal
G02	Movimiento de interpolación Circular a mano derecha (CW)
G03	Movimiento de interpolación Circular a mano izquierda (CCW)
G04	Pausa (Dwell)
G09	Alto o paro total
G10	Fijar desplazamientos
G12	Fresado de cavidad circular a mano derecha
G13	Fresado de cavidad circular a mano izquierda

Cuadro A.1: Tabla de códigos G

CÓDIGO	FUNCIÓN
G17	Selección del plano XY
G20	Seleccionar pulgadas
G21	Seleccionar milímetros
G28	Retorno al cero de la máquina
G29	Retorno al punto de referencia
G31	Saltar, omitir o pasar por alto la función
G35	Medición automática del diámetro de la herramienta
G36	Medición automática de los desplazamientos de trabajo
G37	Medición automática de los desplazamientos de la herramienta
G40	Cancelar compensación del cortador
G41	Compensación del cortador a la izquierda
G42	Compensación del cortador a la derecha
G43	Compensación + de la longitud de la herramienta
G44	Compensación – de la longitud de la herramienta
G47	Grabado de texto
G49	Cancela G43/G44/G143
G50	Cancelar G51
G51	Escalando
G52	Establecer el sistema de coordenadas
G53	Selección fuera de modalidad de las coordenadas de la máquina
G54-G59	Selección del sistema de coordenadas de trabajo
G60	Posicionamiento unidireccional
G61	Alto total dentro de modalidad (Exact Stop modal)
G64	Cancelar G61
G65	Llamada macro-subrutina
G68	Rotación
G69	Cancelar G68
G70	Configuración circular para agujeros para tornillos
G71	Arco modelo para agujeros para tornillos
G72	Agujeros para tornillos a lo largo de un ángulo
G73	Ciclo preprogramado de taladro con avances cortos a alta velocidad
G74	Ciclo preprogramado de roscado inverso
G76	Ciclo preprogramado de acabado fino cilíndrico
G77	Ciclo preprogramado de perforado de la parte posterior cilíndrica

Cuadro A.2: Tabla de códigos G - continuación

CÓDIGO	FUNCIÓN
G80	Cancelar ciclo preprogramado
G81	Ciclo preprogramado de taladro
G82	Ciclo preprogramado de taladrado de centros
G83	Ciclo preprogramado de taladrado en avances cortos
G84	Ciclo preprogramado de roscado o machuelado
G85	Ciclo preprogramado de perforado cilíndrico
G86	Ciclo preprogramado de alto / perforado cilíndrico
G87	Ciclo preprogramado de retracción manual / perforado cilíndrico
G88	Ciclo preprogramado de retracción manual/pausa/perforado cilíndrico
G89	Ciclo preprogramado de pausa y perforado cilíndrico
G90	Sistema absoluto
G91	Sistema incremental
G92	Establecer coordenadas de trabajo
G93	Modalidad de avance de tiempo inverso
G94	Modalidad de avance por minuto
G98	Retorno al punto inicial
G99	Retorno al plano R
G100	Cancelar la imagen espejo
G101	Activar la imagen espejo
G102	Salida programable al RS-232
G103	Límite del previsor de bloques
G107	Correlación o transformación cilíndrica
G110-G129	Seleccionar del sistema de coordenadas de trabajo
G136	Medición automática del centro del desplazamiento de trabajo
G143	Compensación de la longitud de herramienta en el 5 eje
G150	Propósito general de fresado de cavidad
G174-G184	Roscado rígido con propósitos generales
G187	Control de precisión para maquinado en alta velocidad

Cuadro A.3: Tabla de códigos G - continuación 2

CÓDIGOS M	
CÓDIGO	**FUNCIÓN**
M00	Alto al programa
M01	Alto opcional al programa
M02	Fin del programa
M03	Husillo en sentido horario
M04	Husillo en sentido anti-horario
M05	Alto al husillo
M06	Cambio de herramienta
M08	Encender líquido refrigerante
M09	Apagar refrigerante
M10	Embrague del freno del 4º. eje
M11	Desembragar freno del 4º eje
M12	Embrague del freno del 5º.eje
M13	Desembrague del 5º.eje
M16	Cambio de herramienta (igual que M06)
M19	Orientación del husillo
M21-M28	Función pulsada m con aletas para disipar calor
M30	Fin de programa y retorno al inicio del programa
M31	Transportador de virutas hacia delante
M32	Transportador de virutas hacia atrás
M33	Alto al transportador de virutas
M34	Incrementar la posición de la espiga del refrigerante
M35	Disminuir la posición de la espiga del refrigerante
M36	Rotar la paleta
M39	Rotar el carrusel de herramienta
M41	Sobre-controlar el engranaje de baja
M42	Sobre-controlar el engranaje de alta
M51-M58	Encender códigos M opcionales del usuario
M75	Establecer G35 o G136 punto de referencia

Cuadro A.4: Tabla de códigos M

CÓDIGO	FUNCIÓN
M76	Desactivar pantallas
M77	Activar las pantallas
M78	Alarma si se encuentra la señal de omisión
M79	Alarma si no se encuentra la señal de omisión
M82	Soltar o liberar la herramienta
M86	Fijar o sujetar la herramienta
M88	Encender el refrigerante a través del husillo
M89	Apagar el refrigerante a través del husillo
M95	Modalidad de dormir la máquina cuando no esta en uso
M97	Saltar u omitir si no hay señal
M98	Llamada a subprograma local
M99	Retorno o repetición del subprograma

Cuadro A.5: Tabla de códigos M Continuación

Apéndice B

GNU Free Documentation License

Version 1.3, 3 November 2008

http://fsf.org/

Preamble

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work,

regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

B.1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some

widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque". Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text. The "publisher" means any person or entity that distributes copies of the Document to the public. A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. TheseWarranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that theseWarranty Disclaimers may have is void and has no effect on the meaning of this License.

B.2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept

compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

B.3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages. If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

B.4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

1. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if

there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

2. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

3. State on the Title page the name of the publisher of the Modified Version, as the publisher.

4. Preserve all the copyright notices of the Document.

5. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

6. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

7. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

8. Include an unaltered copy of this License.

9. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

10. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

11. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

12. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

13. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

14. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

15. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

B.5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise

combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

B.6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

B.7. AGGREGATION WITH INDEPENDENT-WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

B.8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a

disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

B.9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

B.10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

B.11. RELICENSING

"Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

"CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization. "Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

Copyright c YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with . . . Texts." line with this:

with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software. 161