Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

SIMULADOR DE PROCESO INDUSTRIAL

Proyecto Final de Carrera

Ingeniería Técnica en Informática de Gestión

Autor: Manuel Lora Hermán

Director: Antonio Martí Campoy

Septiembre 2012

Resumen

El presente proyecto, se planteó como una futura implementación en el proyecto realizado en junio de 2007 por Miguel Carro Pellicer, y de título “PFC II-A-DISCA- 55/06 SIMULADOR DE TARJETA DE ADQUISICIÓN DE DATOS “NuDAQ / NuIPC 9112 Series”.

Gracias a este proyecto, se consiguió que los alumnos de prácticas de la asignatura “Informatización Industrial”, dispusieran de una herramienta software que emulara esta tarjeta, lo cual les permitía el poder realizar sus prácticas en cualquier ordenador, tanto dentro como fuera de la UPV, sin necesidad de ningún hardware y sin coste alguno.

Como parte de este proyecto y para demostrar el funcionamiento de esta tarjeta, Miguel Carro realizó una aplicación genérica con la cual se pudiera manejar la tarjeta, activando señales y recibiendo lecturas de esta.

El proyecto que nos atañe pretende pasar de ese proceso genérico, a uno específico que simule un proceso industrial real. Como tal, se ha elegido un proceso de frío industrial en una planta dedicada al almacenamiento de alimentos en cámaras de refrigeración.

Lo que se pretende, es que los alumnos dispongan de una herramienta software en la cual simular las situaciones reales que se dan en una instalación de estas características, y que puedan programar como debe actuar cada elemento, en función de los requisitos y las circunstancias de la planta. Todo ello obviamente, sin coste alguno y sin necesitar acceso a una instalación de este tipo.

Tabla de contenidos

1 INTRODUCCIÓN

El documento que tenemos a continuación sirve para presentar el trabajo realizado en el actual proyecto de final de carrera.

En junio de 2007, Miguel Carro Pellicer presentó su proyecto final de carrera “Simulador de tarjeta de adquisición de Datos ‘NuDAQ / NuIPC 9112 Series’ ”, cuyo objetivo era implementar una serie de herramientas software que simularan el comportamiento de esta tarjeta. Dicho hardware era utilizado en las prácticas de laboratorio de la asignatura “Informatización Industrial”, perteneciente al cuarto curso de la titulación de "Ingeniero Industrial" y que se imparte en la Escuela Técnica Superior de Ingenieros Industriales (ETSII).

Además de usar la tarjeta “NuDAQ / NuIPC 9112 Series”, también requería un emulador hardware de procesos, conectado a dicha tarjeta. Los parámetros de entrada de la tarjeta eran modificados por el alumno a través del proceso y cargaba un programa de prácticas que realizaba tareas de configuración y obtención de datos de la tarjeta utilizando la librería de esta tarjeta.

Dichas prácticas estaban compuestas por los siguientes elementos:

1. Tarjeta de adquisición de datos “NuDAQ / NuIPC 9112 Series”
2. Emulador hardware de procesos
3. Programa de prácticas del alumno
4. Librería de la tarjeta suministrada por el fabricante
5. Compilador de lenguaje C
6. PC

Con todo esto, estaba claro que era muy complicado que el alumno dispusiera de todo este material para practicar en casa, debido a su coste. Por ello, Miguel Carro desarrolló unas herramientas software equivalentes a dicho hardware, quedando de la siguiente forma:

1. La tarjeta “NuDAQ / NulPC 9112 Series” fue simulada por una tarjeta virtual de adquisición de datos.
2. Un simulador software de procesos que permitía cargar y obtener datos de la tarjeta virtual.
3. El programa de prácticas del alumno no sufrió modificaciones, la compatibilidad siguió siendo exacta.
4. La librería de la tarjeta fue modificada, preparada para comunicar con procesos virtuales en lugar del hardware.
5. El mismo compilador y el mismo PC.

Entre las futuras ampliaciones que proponía Miguel Carro en su proyecto, contemplaba la posibilidad de mejorar el proceso virtual existente añadiendo funcionalidades, o basar nuevos procesos virtuales en otras ideas.

2 OBJETIVOS

El objetivo del presente proyecto consiste en desarrollar una aplicación gráfica que simule un proceso industrial, utilizando la tarjeta virtual y su librería, desarrolladas en su proyecto final de carrera por Miguel Carro Pellicer.

Esta aplicación se comunicará única y exclusivamente con la tarjeta virtual de adquisición de datos, enviando y recibiendo información en voltios. La tarjeta virtual convertirá la información a digital y se la pasará a la librería.

La intención es proporcionar a los alumnos un simulador de un proceso industrial, para que puedan realizar prácticas sin necesidad del hardware.

También habrá que programar en lenguaje C una aplicación de consola, la cual será la única que utilice la librería de la tarjeta. Esta aplicación servirá para probar el correcto funcionamiento del simulador, y en el futuro se corresponderá con la aplicación que deberán hacer los alumnos en sus prácticas.

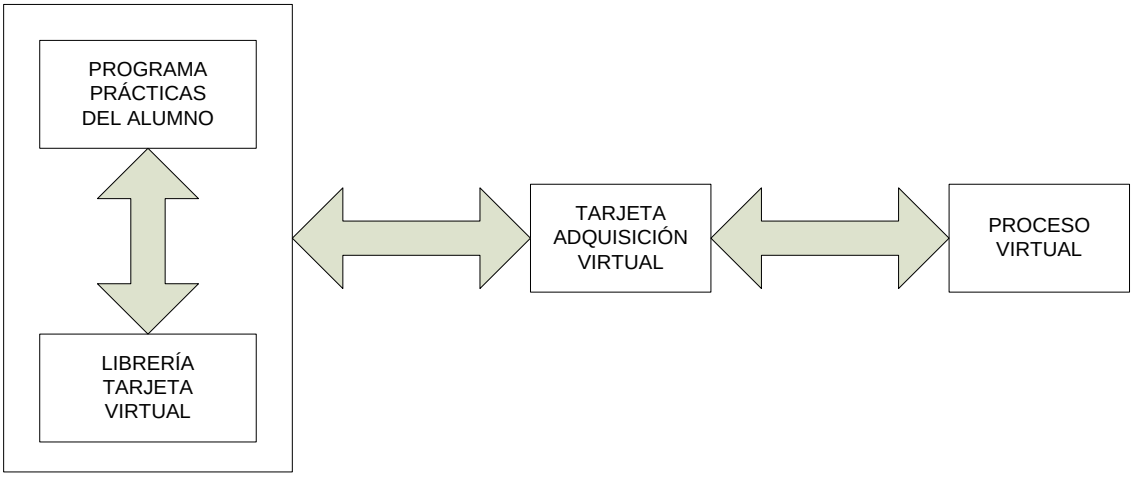
En este caso, se ha elegido un proceso de frío industrial, en el cual se simulará el funcionamiento de una planta industrial, dedicada al almacenamiento de alimentos en cámaras de refrigeración. La intención es que los alumnos dispongan de una herramienta software en la cual simular las situaciones reales que se dan en una instalación de este tipo, y que puedan programar como debe de actuar cada elemento en función de los requisitos y las circunstancias de la planta.

Con todo esto, podemos considerar que los objetivos que se deberán alcanzar serán los siguientes:

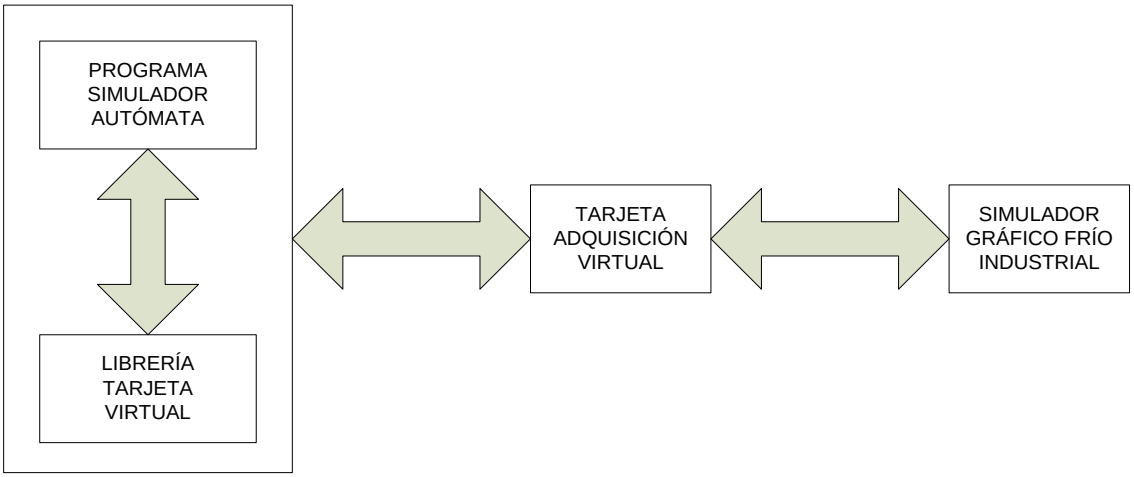
1. Desarrollar una aplicación software que simule un proceso de frío industrial.
    - Será desarrollada en el entorno de desarrollo Borland Delphi 7.
    - Deberá ser sumamente eficiente y más que apta para desempeñar las tareas a las que está destinada.
    - Deberá ser fácil de usar.
    - Deberá ser fácil de instalar.
    - Su funcionamiento deberá de corresponderse al proceso real que simula, salvo en los tiempos de actualización de las temperaturas, con el fin de evitar grandes esperas para ver los resultados de los cambios realizados.
2. Utilizando el lenguaje de programación C, realizar el programa que comunica con la librería de la tarjeta, el cual nos servirá para probar el correcto funcionamiento del simulador.

3. Cumplir con los plazos de entrega previstos.

El proyecto de Miguel Carro quedaba representado de la siguiente forma:



El actual proyecto, lo que hará es reemplazar ese "Proceso virtual" genérico, por un "Simulador gráfico de frío industrial", y para la presentación del mismo y la correspondiente comprobación de que todo funciona como debe, también se entregará un programa similar al que realizarán los alumnos en sus prácticas, y etiquetado en el diagrama como "Programa prácticas del alumno", y que en nuestro caso simula el programa de autómata que habría en una instalación real.



3 ANÁLISIS DE REQUISITOS

Se ha elegido un proceso de frío industrial, en el cual se simulará el funcionamiento de una planta industrial, dedicada al almacenamiento de alimentos en cámaras de refrigeración. En ella deberemos encontrar los diferentes elementos necesarios para obtener el frío que se precisa para conservar dichos alimentos. Será posible interactuar con estos elementos, con el fin de que el alumno pueda simular las distintas situaciones que se pueden dar en la vida real en un proceso de este tipo, el cual se ve afectado directamente por los cambios de temperatura ambiente, averías en alguno de los equipos, así como la acción directa de los operarios de dicha planta, los cuales abren y cierran las cámaras, cambian las consignas de temperatura de trabajo de las cámaras, en función de los alimentos que van a contener, las llenan y las vacían, etc. De esta forma, lo que se deberá de programar es que al actuar sobre los elementos de la instalación, este actualizará el dato correspondiente en la tarjeta virtual, enviándole el mensaje con la información necesaria para ello. Del mismo modo, la aplicación estará en todo momento a la espera de recibir mensajes, para actualizar si debe, el elemento asignado a la salida digital o analógica correspondiente.

En este tipo de instalaciones, en el cuadro eléctrico que las controla suele haber un autómata o controlador, que es el encargado de controlar los estados en los que se encuentran cada uno de los elementos, así como de recibir las lecturas de las sondas de temperatura, y en función de esto, y de los parámetros programados por los operarios, hacer que se activen las señales necesarias para que se pongan en marcha en cada momento, los elementos necesarios para que todo funcione como debe. En nuestro caso, la labor de ese autómata se corresponde con el programa en modo consola de C que debemos hacer, y en el cual se recibirán las lecturas tanto digitales como analógicas, y en función de lo deseado, haremos que se activen las salidas digitales correspondientes y se actualicen las salidas analógicas. Para esto, utilizaremos las llamadas a la librería disponibles.

Este programa se realizará para poder probar el correcto funcionamiento del simulador, y en el futuro les corresponderá a los alumnos el realizarlo en sus correspondientes prácticas.

4 DISEÑO DEL PROYECTO

Para desarrollar la aplicación del simulador, se ha utilizado el lenguaje Object Pascal, dentro del entorno de programación Borland Delphi 7, el cual es un entorno de desarrollo flexible y potente. Una de las mayores ventajas de la programación en Delphi es que es una programación orientada a objeto, lo cual facilita mucho la labor del programador.

Podría haberse realizado en Borland CBuilder C++, por utilizar el mismo lenguaje C utilizado en el programa que simula el autómata, pero no había problema en utilizar distintos lenguajes, y tengo más experiencia en el desarrollo de aplicaciones en el entorno de Delphi.

En el programa que equivale al que tendrán que realizar los alumnos en sus prácticas, si que se ha utilizado el lenguaje C, puesto que la librería de la tarjeta virtual está programada en este lenguaje, y para no tener problema de incompatibilidades lo hacía lo más idóneo. Otro tema a tener en cuenta, es el hecho de que los alumnos trabajaran en este lenguaje.

4.1 Mecanismos de intercomunicación de procesos

Tal como se ha comentado anteriormente, este proyecto se basa en el presentado por Miguel Carro en junio de 2007. Esto hace que existan ciertas decisiones que él tuvo que tomar en su momento a la hora de realizar de una u otra forma la aplicación, y que ahora ya no tiene sentido volverse a plantear, o incluso serían inviables hacerlas de otro modo, para que todo funcione correctamente.

Uno de estos temas es el mecanismo de intercomunicación de procesos. En su momento, se decidió utilizar “pipes” del sistema operativo, y tanto la tarjeta virtual como la librería, utilizan estos, por lo que tanto en la aplicación del simulador, como en el programa del proceso, se ha tenido que continuar con esta opción.

En primer lugar detallaremos la estructura de las tramas utilizadas. Como ya hemos indicado, esta información nos viene dada del proyecto de Miguel Carro:

[CÓDIGO_OPERACIÓN # DIRECCIÓN o CANAL # VALOR]

Se trata de una cadena de como máximo 20 caracteres, en la cual se han separado sus 3 elementos por el carácter ‘#’. Estos 3 elementos son:

1. **Código_operación**: Indica el código de la función de la librería que ha enviado dicha trama. Tenemos las siguientes opciones:

    - "***DO***": La llamada que ha enviado la trama es DO_WritePort indicándole mediante este código de operación a la tarjeta la función que debe realizar. La tarjeta virtual genera una trama con este código cuando quiere cambiar el estado de alguno de los dispositivos de la aplicación gráfica. Se trata de una trama que contendrá el estado de las salidas digitales de la tarjeta. El campo "Valor" de la trama contendrá un número entero, en el que sus bits representan cada una de estas salidas digitales.
    - "***AO***": La llamada que ha enviado la trama es AO_WriteChannel indicándole mediante este código de operación a la tarjeta la función que debe realizar. La aplicación gráfica recibe una trama desde la tarjeta virtual con este código, cuando en esta se ha actualizado el valor de una salida analógica, cuyo valor hay que refrescar en la aplicación.
    - "***DI***": La llamada que ha enviado la trama es DI_ReadPort indicándole mediante este código de operación a la tarjeta la función que debe realizar. A través de una trama con este código, el proceso que simula el programa del autómata recibe de la tarjeta virtual, el valor que contienen las entradas digitales. El valor recibido será un entero cuyos bits representan cada una de estas entradas digitales.
    - "***AI***": La llamada que ha enviado la trama es AI_ReadChannel indicándole mediante este código de operación a la tarjeta la función que debe realizar. A través de una trama con este código, el proceso que simula el programa del autómata recibe de la tarjeta virtual, el valor que contienen las entradas analógicas.
    - "***XX***": Una trama con este código de operación, indica a la tarjeta que debe actualizar en sus registros el valor digital, ya que se ha cambiado esta entrada. Será la aplicación gráfica la que utilice una trama con este código para enviar esta actualización a la tarjeta virtual.
    - "***XZ***": Una trama con este código de operación, indica a la tarjeta que debe actualizar en sus registros el valor analógico, ya que se ha cambiado esta entrada. Será la aplicación gráfica la que utilice una trama con este código para enviar esta actualización a la tarjeta virtual.

2. **Dirección o Canal**: En las operaciones digitales servirá para indicar el número de puerto y en las operaciones analógicas el número de canal.

3. **Valor**: Determina el campo de datos de la trama. Cuando se trata de una trama que actualiza un valor digital, este campo "Valor" contendrá un número entero, cuyos bits representarán cada una de las salidas o entradas digitales. En el caso de las tramas para actualizar valores analógicos, este campo "Valor" contendrá un número en formato decimal, salvo en el caso de las tramas con el código "AI", donde el valor será un número entero que habrá que escalar adecuadamente para obtener el número decimal esperado. En todos los casos se tratará de valores de un tamaño de 4 bytes.

En este tipo de tramas, es fácil codificar y decodificar el mensaje, teniendo en cuenta que el separador determina claramente cada uno de los datos enviados o recibidos.

Ahora que ya conocemos la estructura de la trama, podemos ver cómo está diseñado el protocolo de comunicación y que elementos intervienen. Dado que el proyecto que nos atañe sólo utiliza la tarjeta virtual así como su librería, pero no precisa de ningún modo el modificarlas, tendremos en cuenta que obviamente, ambas intervienen en estas comunicaciones, pero no entraremos en el detalle de los elementos y funciones que las componen.

- Proceso de frío industrial

  - Hilo de programación llamado "Hilo Datos Salida".
    Este hilo será el responsable de esperar tramas procedentes de la tarjeta virtual que indiquen que debe realizarse una operación de salida digital o analógica. Se conectará a la tarjeta a través del tubo "tubotest" que comentaremos posteriormente y esperará la recepción de una trama a través del evento "EventoRecibir" que comentaremos posteriormente.
  - Tubo con nombre "tubotest".
    Este tubo será el medio de comunicación para recibir las tramas procedentes de la tarjeta que indiquen operaciones de salida. Se utilizará de modo unidireccional, de la tarjeta virtual al proceso virtual. Las tramas serán esperadas por el hilo "Hilo Datos Salida" comentado anteriormente.
  - Tubo con nombre "tubotest2".
    Este tubo será el medio de comunicación para actualizar datos en la tarjeta. Cada vez que se cambie una entrada en el proceso virtual, éste debe enviar los datos actualizados a la tarjeta. Para ello envía a través de este tubo la trama, y conectará con el hilo servidor de actualizaciones de la tarjeta al que llamaremos "Hilo Servidor Actualizaciones". Se utilizará de modo unidireccional, del proceso virtual a la tarjeta virtual.

- Evento de nombre “EventoRecibir”.

  Este evento se utiliza por motivos de sincronización, cada operación de envío de una trama con operaciones de salida será notificada por la tarjeta virtual. El hilo anteriormente mencionado “Hilo Datos Salida”, dedicado a recibir tramas de la tarjeta, será el encargado de esperar a que se de este evento, suspendiéndose mientras no se de el evento.

- Evento de nombre “EventoEnviar”.

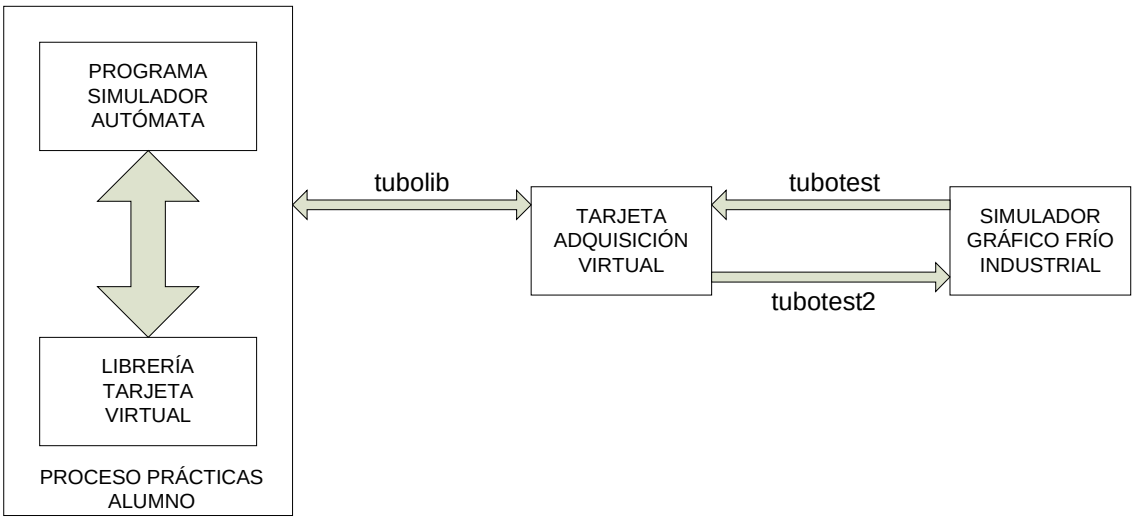
  Este evento se utiliza por motivos de sincronización. Cada vez que se actualiza un dato en el proceso virtual, éste debe enviar una trama de actualización a la tarjeta virtual. El evento suspenderá al hilo “Hilo Servidor de Actualizaciones” que comentaremos posteriormente. El proceso virtual notificará mediante este evento, que va a enviar una actualización de un dato en una trama, a través del tubo “tubotest2”.

- Tarjeta virtual

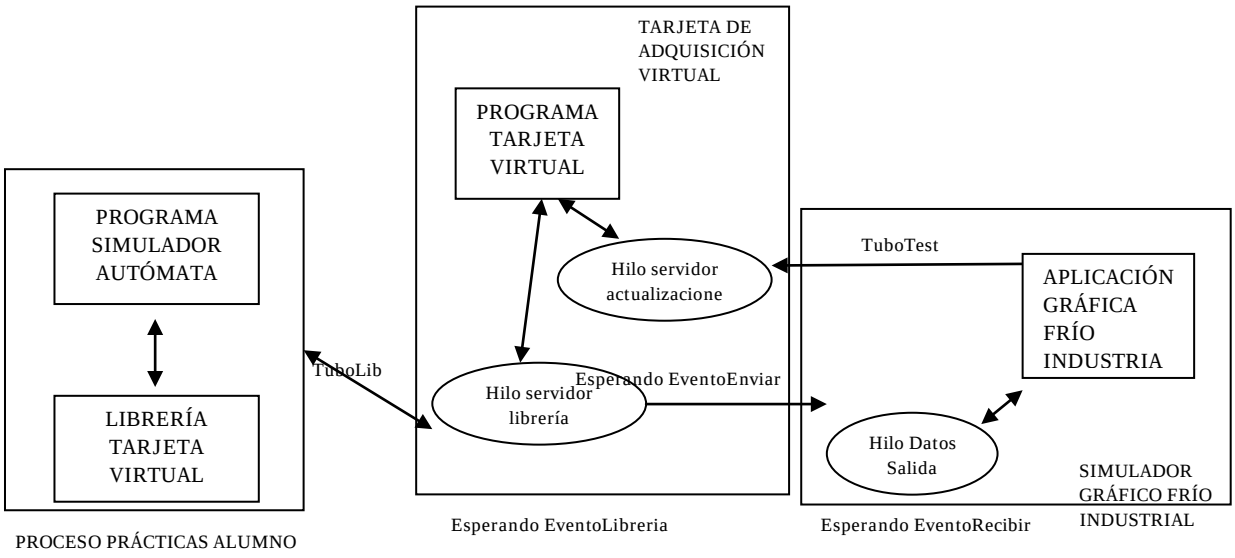
- Librería tarjeta

Es necesario utilizar 2 tubos entre el proceso virtual y la tarjeta virtual, para evitar que se bloquee la aplicación, en el caso de que se superara el tamaño máximo del buffer, cosa que podría ocurrir, puesto que se realizan accesos a alta velocidad en ambos sentidos. Destinando un tubo a cada uno de estos procesos, evitamos este posible colapso. Para las comunicaciones entre la librería y la tarjeta, esto no es necesario porque las llamadas son completamente sincronizadas.

La situación queda de la siguiente forma:



Si añadimos los hilos de programación y los eventos, el esquema quedaría de la siguiente forma:



4.2 Capa lógica de la aplicación

Al hablar de la capa lógica, estamos hablando de cómo hemos planteado la lógica del programa, su estructura y los algoritmos de los protocolos utilizados. También tendremos en cuenta que variables emplearemos para almacenar la información.

Hemos utilizado variables para guardar los manejadores de los tubos, los identificadores de los eventos y por supuesto para guardar los datos que se utilizan en los procedimientos y funciones encargados de las comunicaciones con la tarjeta.

Por otro lado, hemos definido variables, para guardar los valores que se reciben de la tarjeta virtual y que posteriormente nos servirán para saber que imágenes debemos mostrar en cada momento.

También hemos creado otras variables, para saber en todo momento en qué estado se encuentran ciertos elementos que son manipulados en la aplicación, bien sea por el propio usuario, como por ejemplo los diferenciales del cuadro eléctrico, o de forma automática, como las temperaturas de la cámara y el glicol.

Otras variables que hemos utilizado, nos han servido para saber en cada momento que imagen tenemos cargada en pantalla. Esto nos servirá para poder realizar las animaciones de estas imágenes.

En la aplicación hemos definido un hilo, el cual se encargará de recibir las tramas que son enviadas desde la tarjeta virtual. Este hilo tendrá la siguiente estructura:

a. Abrir evento para recibir datos de la tarjeta.
b. Esperar evento
c. Evento Recibido
d. Leer datos de la tarjeta
e. Decodificar orden
f. Mostrar datos en el lugar correspondiente.

Dado que el dato que se envía y se recibe de la tarjeta virtual, viene en formato de entero, pero en la aplicación se trabaja con los bits individuales, se han definido funciones que se encargan de realizar estas conversiones.

Como ya es sabido, para el presente proyecto no tiene sentido que se ejecute la aplicación, sin que se ejecute la tarjeta, por ello cuando se crea esta, automáticamente se crea una tarjeta virtual asociada. De este modo, la tarjeta virtual será un proceso hijo de la aplicación y se comunicarán a través de los tubos.

En la aplicación se ha pretendido que los elementos que intervienen en el proceso del frío, motor del compresor, ventiladores, bombas, etc., tengan un cierto grado de animación para que sea más visual la aplicación, y se tenga claro en todo momento su situación. Para realizar esto, no es posible simplemente actualizar su estado en función de la lectura de la tarjeta. Por ello, necesitamos un temporizador para que teniendo en cuenta el estado de cada elemento, alterne la carga de las imágenes, para simular movimientos. Para cada elemento, definiremos un procedimiento encargado de simular su estado, y el temporizador será el encargado de llamar a estos procedimientos. La estructura que tendrán estos procedimientos será:

a. Comprobar el bit de marcha del elemento
b. Si está en marcha, comprobar la imagen cargada y cargar la alternativa.
c. Si no está en marcha, comprobar si está en avería.
d. Si está en avería, comprobar la imagen cargada y cargar la alternativa para mostrar el estado de avería.
e. Si no está en marcha ni en avería, cargar la imagen de parado.

Otro punto delicado de la aplicación, es el simulado de los cambios de temperatura en el tanque de glicol y de la cámara. En la vida real, simplemente hay una sonda que recoge estas temperaturas y envía esa información al autómata, en nuestro caso la tarjeta virtual, y con esta información y los

parámetros de trabajo definidos actúa en consecuencia. En nuestro caso, debemos hacer que dependiendo de las circunstancias que se den en la aplicación, se produzca esa variación de temperatura y que posteriormente esta se envíe a la tarjeta como si fuera una lectura de la sonda. Para realizar esto, pondremos un temporizador que tenga en cuenta todas esas circunstancias y realice un cálculo aproximado de cómo podría variar esas temperaturas. Para hacer más ágil la aplicación se acortarán los tiempos de respuesta, ya que de hacerlo a tiempo real, se necesitaría mucho tiempo para ver la simulación.

Para la temperatura de la cámara, este valor se calculará en función de la temperatura ambiente, de si está en marcha el ventilador de la cámara, de si está llena de género y si la puerta está abierta o cerrada.

En el caso de la temperatura del glicol, este valor se calculará en función de la temperatura ambiente y de si está en marcha el circuito primario, controlado en este caso simplemente mirando si está en marcha el compresor.

4.3 Capa de la interfaz de usuario

Se ha pretendido que el usuario tenga una clara una visión de los elementos que constituyen este tipo de instalaciones. Para ello, se han empleado dibujos realizados con detalle, basados en imágenes reales de cada uno de los equipos que intervienen en una planta de frío.

Para empezar tendremos un cuadro eléctrico, en el cual habrá elementos en sus puertas y otros en su interior, para lo cual se ha dibujado con una de sus puertas abiertas. En la puerta visible tendremos un interruptor general, encargado de dar o quitar tensión a toda la planta. Al igual que ocurre en una instalación real, también existirá un selector que nos permita habilitar o no la cámara.

En cuanto al interior, dispondremos de una serie de disyuntores que nos permitirán simular averías en cada uno de los equipos que se encargan de conseguir el frío que se precisa en la cámara. Para tener claro en qué estado se encuentra cada uno, cuando estén activos estarán pintados de color verde y en caso contrario estarán en rojo.

Entre los equipos que representaremos tendremos un compresor, encargado de comprimir el gas que circula por el circuito. Para hacerlo más visual el motor de este cambiará de color en función del estado en el que se encuentre. Cuando esté parado será de color gris, cuando esté averiado será de color rojo y cuando esté en funcionamiento alternará entre dos tonos de verde.

El siguiente elemento que nos encontramos en el circuito es el condensador. Este equipo precisa de unos ventiladores y una bomba de agua,

para condensar el gas que recibe del compresor. Al igual que ocurría con el compresor, tanto los ventiladores como la bomba cambiarán de color en función de su estado y se utilizará el mismo criterio de colores que para el compresor. En el caso de los ventiladores, cuando estén en marcha, se verán girar las aspas, haciendo más real el proceso.

A continuación tendremos el separador de amoniaco. En él tendremos una válvula de expansión, encargada de que el gas en estado líquido que se recibe del condensador, cambie de nuevo a estado gaseoso y por este cambio de estado se consiga que se enfríe drásticamente. Al tratarse de un circuito cerrado permanecerá en estado medio líquido medio gaseoso en el depósito. En este separador también podremos ver dibujado el intercambiador de placas, encargado de que este amoníaco líquido y superfrío, enfríe el agua glicolada que viene de su depósito y que se hace pasar por este elemento antes de recuperarla de nuevo en su depósito.

Siguiendo el circuito llegaremos al depósito de glicol, en el cual tendremos un visualizador de su temperatura, así como 2 bombas, una en el circuito primario y otra en el secundario. La primera se encargará de hacer circular el agua glicolada por las placas del intercambiador para que esta se enfríe y la segunda se encargará de hacer circular esta agua fría por el circuito que llega hasta la cámara, para hacer llegar el frío a esta. Ambas bombas seguirán el mismo criterio de estados y colores que el resto de elementos explicados, para simular su funcionamiento.

Para finalizar el circuito tendremos la cámara, encargada de almacenar el género que se desea enfriar. En esta tendremos unos ventiladores encargados de hacer circular aire entre el circuito de tuberías con agua fría que llegan desde el depósito de glicol, haciendo que este aire frío entre en la cámara. Además de esto tendremos opción de abrir y cerrar la puerta, puesto que el estado de esta influirá a la hora de simular la variación de temperatura dentro de la cámara. En el interior se ha colocado una seta de emergencia para avisar de que hay un hombre encerrado. Si esta es pulsada y la puerta está realmente cerrada, se activará una sirena en el exterior de la cámara para avisar de esto.

Para el control de esta cámara se ha colocado en un lateral un control de temperatura. En este equipo podremos ver tanto la temperatura real de la cámara, como la consigna de marcha o la consigna de paro, según se haya seleccionado. Para realizar esta selección tenemos unos botones a la izquierda que nos permitirán cambiar esto y a la derecha tendremos otros, para subir o bajar de temperatura de las consignas de marcha y paro.

Para simular el llenado de la cámara se ha situado un botón, con el cual podremos llenar o vaciar la cámara, según corresponda.

Por último, tendremos un control para simular los cambios de temperatura ambiente, la cual afecta directamente a la simulación de temperatura tanto en la cámara como en el glicol.

4.4 Tabla de entradas y salidas

Para cada uno de los elementos de la aplicación que interactúan con la tarjeta virtual, debemos de asignarle su correspondiente bit o canal de la tarjeta. Esta relación la mostramos a continuación, indicando además su rango o el significado para cada uno de los estados, en el caso de las entradas y salidas digitales.

Entradas digitales		
Bit	Señal	Estados
0	Interruptor general del cuadro eléctrico	1 - Conectado / 0 - Desconectado
1	Diferencial del compresor	1 - Ok / 0 - Avería
2	Diferencial del ventilador del condensador	1 - Ok / 0 - Avería
3	Diferencial de la bomba del condensador	1 - Ok / 0 - Avería
4	Diferencial de la bomba del circuito primario	1 - Ok / 0 - Avería
5	Diferencial de la bomba del circuito secundario	1 - Ok / 0 - Avería
6	Diferencial del ventilador de la cámara	1 - Ok / 0 - Avería
7	Sensor de puerta abierta	1 - Abierta / 0 - Cerrada
8	Seta de emergencia de hombre encerrado en cámara	1 - Activada / 0 - No activada
9	Paro/marcha de la cámara	1 - Marcha / 0 - Paro
Salidas digitales		
Bit	Señal	Estados
0	Marcha compresor	1 - Marcha / 0 - Parado
1	Marcha ventilador del condensador	1 - Marcha / 0 - Parado
2	Marcha bomba del condensador	1 - Marcha / 0 - Parado
3	Marcha bomba del circuito primario	1 - Marcha / 0 - Parado
4	Marcha bomba del circuito secundario	1 - Marcha / 0 - Parado
5	Marcha ventiladores de la cámara	1 - Marcha / 0 - Parado
Entradas analógicas		
Canal	Señal	Rango
0	Consigna de paro de la cámara	-3ºC .. 40ºC
1	Consigna de marcha de la cámara	-2ºC .. 40ºC
2	Temperatura de la cámara	-5,75ºC .. 40ºC
3	Temperatura del depósito de glicol	-12,75ºC .. 40ºC

Hay que tener en cuenta que se ha simulado la lectura de temperatura utilizando unas sondas de temperatura que leen valores entre -40°C y 60°C y que habrá que realizar el correspondiente escalado.

5 IMPLEMENTACIÓN

5.1 Mecanismos de intercomunicación de procesos

Tal como se indicó en la parte de diseño, el presente proyecto requiere el desarrollo de una aplicación que representa gráficamente un proceso industrial. Tenemos claro que en este desarrollo, estamos utilizando la tarjeta virtual y su correspondiente librería, pero no es preciso entrar en el detalle de su funcionamiento, puesto que esto está claramente expuesto en el proyecto de Miguel Carro, desarrollador de este software.

Por ello, en esta parte, entraremos en el detalle de las funciones que intervienen en la parte del simulador gráfico, y obviaremos la parte concerniente a la tarjeta y la librería.

```
BOOL CreateProcess(
        LPCTSTR lpApplicationName,
        LPTSTR lpCommandLine,
        LPSECURITY_ATTRIBUTES lpProcessAttributes,
        LPSECURITY_ATTRIBUTES lpThreadAttributes,
        BOOL bInheritHandles,
        DWORD dwCreationFlags,
        LPVOID lpEnvironment,
        LPCTSTR lpCurrentDirectory,
        LPSTARTUPINFO lpStartupInfo,
        LPPROCESS_INFORMATION lpProcessInformation );
```

Utilizamos CreateProcess() para crear un nuevo proceso para la tarjeta virtual. De todos sus parámetros, el único que nos importa, es lpCommandLine, ya que es aquí donde especificamos que vamos a ejecutar la tarjeta. Esta tarjeta virtual, al ser creada desde la aplicación del simulador, será considerada un proceso hijo de este.

```
DWORD WaitForSingleObject(
        HANDLE hHandle,
        DWORD dwMilliseconds);
```

Utilizamos WaitForSingleObject() para esperar el evento "EventoRecibir", pasado como primer parámetro hHandle. Pasaremos como segundo parámetro dwMilliseconds, "INFINITE" para que el intervalo de tiempo de espera nunca transcurra.

```
HANDLE CreateThread(
        LPSECURITY_ATTRIBUTES lpThreadAttributes,
        DWORD dwStackSize,
```

```
    LPTHREAD_START_ROUTINE lpStartAddress,
    LPVOID lpParameter,
    DWORD dwCreationFlags,
    LPDWORD lpThreadId );
```

Con esta función **CreateThread**, crearemos un nuevo hilo pasando como parámetro lpStartAddress la función "ServidorPeticiones" que debe ejecutar, y como parámetro lpThreadId, el identificador del hilo. Este hilo, será el encargado de estar pendiente a la información que transmita la tarjeta virtual.

```
BOOL WaitNamedPipe(
    LPCTSTR lpNamedPipeName,
    DWORD nTimeOut );
```

Utilizaremos esta función **WaitNamedPipe**, para conectar con los tubos "tubotest" y "tubotest2", pasados en el parámetro lpNamedPipeName, que previamente se habrán creado en la tarjeta virtual. Si no se produce ningún error, la comunicación con la tarjeta estará establecida.

```
HANDLE CreateFile(
    LPCTSTR lpFileName,
    DWORD dwDesiredAccess,
    DWORD dwShareMode,
     LPSECURITY_ATTRIBUTES lpSecurityAttributes,
    DWORD dwCreationDistribution,
    DWORD dwFlagsAndAttributes,
    HANDLE hTemplateFile );
```

Con la función **CreateFile**, abriremos el tubo pasado como parámetro lpFileName, como si fuera un fichero. Esta función será llamada, después de haber ejecutado **WaitNamedPipe**, ya que después de crear el tubo, hay que abrir el descriptor del fichero para poder escribir y leer en él. En la llamada, pasaremos como parámetro dwDesiredAccess, "GENERIC_READ or GENERIC_WRITE" y como parámetro dwCreationDistribution, "OPEN_EXISTING", para indicar que queremos abrir el tubo ya existente, en modo lectura y escritura.

```
BOOL ReadFile(
    HANDLE hFile,
    LPVOID lpBuffer,
    DWORD nNumberOfBytesToRead,
    LPDWORD lpNumberOfBytesRead,
    LPOVERLAPPED lpOverlapped );
```

Con esta función **ReadFile**, obtenemos del tubo indicado en el parámetro hFile, una cadena de caracteres de una tamaño especificado en el parámetro nNumberOfBytesToRead, y que se guardan en la variable pasada como parámetro

lpBuffer, devolviendo el número de bytes leidos, en la variable pasada como parámetro lpNumberOfBytesRead.

```
BOOL WriteFile(
        HANDLE hFile,
        LPCVOID lpBuffer,
        DWORD nNumberOfBytesToWrite,
        LPDWORD lpNumberOfBytesWritten,
        LPOVERLAPPED lpOverlapped );
```

La función WriteFile, la utilizaremos para escribir en el tubo indicado en el parámetro hFile, la cadena de caracteres pasados como parámetro lpBuffer, y esta devolverá en el parámetro de salida lpNumberOfBytesWritten, el número de bytes que se han escrito.

```
HANDLE CreateEvent(
        LPSECURITY_ATTRIBUTES lpEventAttributes,
        BOOL bManualReset,
        BOOL bInitialState,
        LPCTSTR lpName);
```

Con la función **CreateEvent**, creamos los eventos "EventoEnviar" y "EventoRecibir", pasando como parámetro lpName, cada uno de estos, y quedando referenciados ambos por su manejador. Estos eventos son unos objetos de sincronización que nos permitirán mantenernos a la espera, hasta que este evento se produzca. En nuestro caso, el "EventoRecibir" lo activarán la librería y la tarjeta, mientras que el "EventoEnviar" lo activaremos desde la aplicación, cada vez que necesitemos actualizar sobre la tarjeta alguna entrada digital o analógica.

```
BOOL SetEvent(
   HANDLE hEvent );
```

Utilizamos la función SetEvent, pasando como parámetro hEvent, el id del "EventoEnviar", cuando vamos a escribir en el tubo, para que la tarjeta sepa que debe leer lo que se escriba en este.

5.2 Capa lógica de la aplicación

En la capa lógica del apartado del diseño, ya se ha detallado con bastante exactitud el funcionamiento de la aplicación. A nivel de implementación, habría que comentar que se han utilizado funciones del lenguaje de programación pascal, dentro del entorno de desarrollo Borland Delphi 7, por lo que tampoco será necesario una explicación completa de cómo está estructurado.

Podemos comentar que para la simulación de las variaciones de temperatura del glicol y la cámara, y para la animación de las imágenes, se han utilizado 2 componentes TTimer.

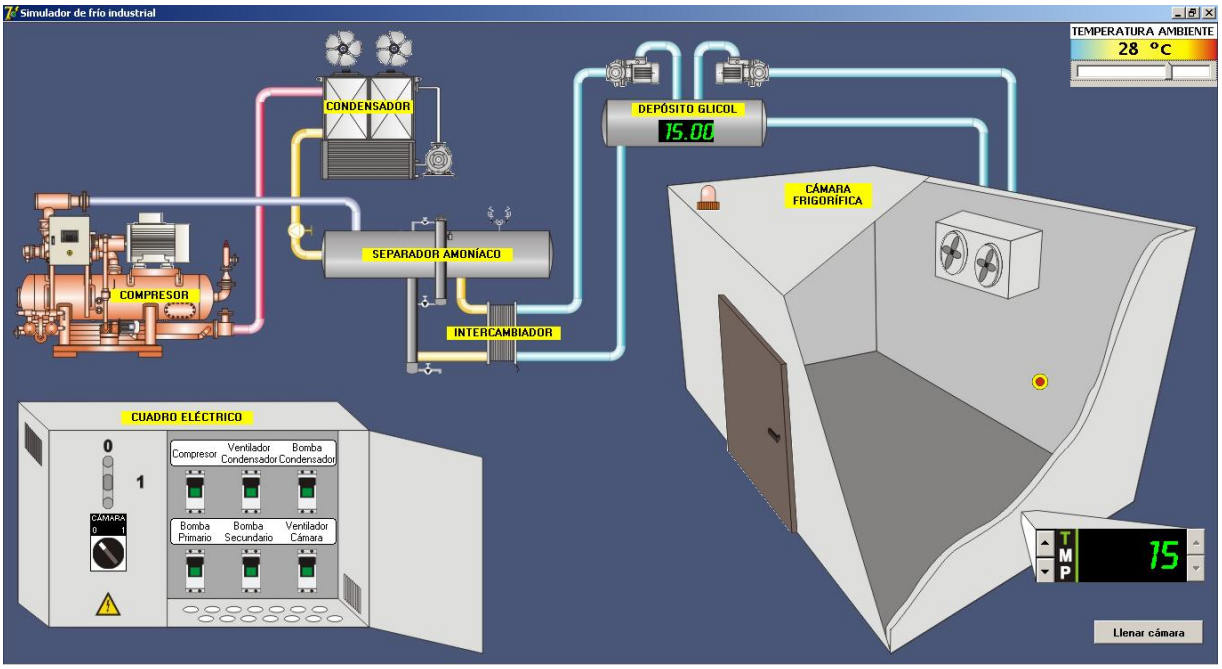
Por un lado, tenemos uno para controlar los estados de los diferentes elementos de la instalación, compresor, ventiladores, bombas y sirena de "hombre encerrado". Este se ejecuta cada segundo, y en su evento "OnTimer", encontramos las llamadas a cada una de los procedimientos que hemos definido para controlar cada uno de los elementos de la instalación, por ejemplo "EstadoCompresor", "EstadoVentiladorCondensador" o "EstadoBombaPrimario". Dentro de cada uno de estos procedimientos, en función de los valores que se han leído de la tarjeta, por ejemplo marcha y avería, se hará que se cargue la imagen adecuada a cada situación, y el caso de que estén en marcha, incluso se podrá simular que los ventiladores giran o que la sirena cambia de color.

Por otro lado, también tenemos otro temporizador, que se ejecuta cada 3 segundos, en el cual se han puesto las llamadas a los procedimientos "TemperaturaCamara" y "TemperaturaGlicol", encargados de simular la variación de temperatura en estos elementos.

5.3 Capa de la interfaz de usuario

Como ya se ha mencionado anteriormente, dentro de este proyecto, la única aplicación que presenta una interfaz gráfica es el "Simulador de frío industrial", la cual como también se ha indicado en alguna ocasión, ha sido realizada en el entorno de desarrollo Borland Delphi 7.

Aquí podemos ver como se presenta dicha aplicación al usuario:



Para realizar esta aplicación, se han utilizado los siguientes objetos de Borland Delphi:

- **TLabel y TStaticText**: Son 2 elementos utilizados para mostrar una cadena de texto en pantalla. En ambos, se utiliza la propiedad "Caption" para indicar el texto que se desea mostrar. Este valor puede fijarse en tiempo de diseño, como en el caso de los letreros que identifican los distintos elementos de la instalación, o también son modificados en tiempo de ejecución, como los utilizados para mostrar las temperaturas de la cámara, del tanque de glicol o la temperatura ambiente.

  En el caso de la temperatura de la cámara y del glicol, este valor se actualizará cada 3 segundos, ya que existe un temporizador "TActualizaTemperatura", en cuyo evento "OnTimer" se han puesto las llamadas a los procedimientos "TemperaturaCamara" y "TemperaturaGlicol", los cuales se encargan de simular estos cambios.

- **TImage**: Este es el componente que en mayor cantidad se ha utilizado en esta aplicación. Utilizamos este objeto para mostrar en pantalla, una imagen cargada desde un fichero. Se han utilizado imágenes en formato .jpg, como la imagen de la planta, o en formato .bmp, como las utilizadas para mostrar los distintos elementos de la instalación que varían en función de los valores recibidos de la tarjeta. El utilizar el formato .bmp nos permite que estas imágenes puedan situarse sobre la imagen de la planta, y utilizando la propiedad "Transparent", que sólo sea visible la imagen deseada, sin mostrar el fondo de dicha imagen.

Para cada uno de estos elementos se ha definido un procedimiento, encargado de hacer que en cada momento se esté mostrando en la imagen en cuestión el fichero de imagen adecuado. Por ejemplo, tenemos un procedimiento llamado “EstadoCompresor”, el cual comprueba los valores de las variables “marcha_compresor” y “averia_compresor”, y utilizando el evento “LoadFromFile” de la propiedad “Picture”, carga la imagen del compresor pintado en verde claro y oscuro (“Motor verde claro.bmp” y “Motor verde oscuro.bmp”) cuando está en marcha, gris (“Motor gris.bmp”) cuando está parado y rojo (“Motor rojo.bmp”) cuando está averiado. Este procedimiento es llamado por un temporizador, cada segundo.

Por otro lado, tenemos otras imágenes como la puerta de la cámara, el interruptor general del cuadro, el selector de habilitado de la cámara o los diferenciales del cuadro, los cuales no varían con el tiempo, sino que es el propio usuario, el que haciendo click sobre estas imágenes, hará que se intercambien las imágenes en cuestión. Por ejemplo, para la puerta de la cámara, se cargarán las imágenes “Puerta abierta.bmp” o “Puerta cerrada.bmp”, en función del estado en el que se encuentren. Cada una de estas imágenes, tienen un evento “OnClick”, en el cual se ha puesto el código necesario para que en función del estado en el que se encuentre el elemento en cuestión, cambie al contrario y actualice la imagen. Además, dentro de la variable de tipo entero “entrada_digital”, se actualizará el bit, correspondiente al elemento sobre el cual se ha pulsado, y esta variable se escribirá en la tarjeta virtual.

En el caso de la imagen del interruptor general, al pulsarla, simularemos que damos o quitamos tensión al cuadro, sin la cual no es posible que funcione ningún elemento de la instalación.

Con el selector de la cámara, indicaremos que la cámara va a necesitar o no frío.

Con los diferenciales de los elementos encargados de hacer el frío, podremos simular que se producen averías eléctricas en estos elementos, lo cual produce que los diferenciales se disparen. El efecto visual que se conseguirá, es que el elemento en cuestión cambie su imagen por una pintada en rojo, y por otro lado, que se paren el resto de elementos que dependen de este, es decir, si se para el compresor, el motor de este se mostrará en rojo y los elementos del condensador, el separador y la bomba del circuito primario de glicol, se mostrarán en gris. Obviamente, al pulsar sobre la imagen, lo único que haremos será escribir en la tarjeta virtual, el bit que indica esta avería. Será el programa que simula ser un

autómata (el que será el programa de prácticas de los alumnos), el encargado de activar los bits necesarios para que la aplicación sepa en qué estado se encuentra cada elemento.

También hemos utilizado una imagen para colocar una "Seta de emergencia", dentro de la cámara, para poder activar la señal de "hombre_encerrado", utilizada para saber dentro del programa que simula el autómata, junto con la señal de "puerta_abierta", si se debe activar la señal que activa la sirena de la cámara, la cual es otra imagen que funciona de forma similar a lo explicado para el compresor.

Por último, también tenemos otra imagen para mostrar o quitar género dentro de la cámara. Esta imagen, se hará visible o no, utilizando su propiedad "Visible", dentro del evento "OnClick" del botón creado para tal fin. Al igual que pasaba con otros elementos, también se actualizará el bit asignado para esto, dentro de la variable "entrada_digital" y esta será escrita en la tarjeta virtual.

- **TBitBtn**: Se ha empleado este objeto para simular el llenado y vaciado de la cámara. Utilizando su evento "OnClick", haremos que en función del estado actual de la cámara, llena o vacía, se haga visible o no la imagen del género, y a su vez, si se está llenando, que la temperatura de la cámara suba 10ºC al haber metido género caliente dentro de la cámara. Cuando cambiamos el estado de la cámara, actualizamos el valor de la propiedad "Caption" del botón, para que muestre que operación realizaremos si lo pulsamos de nuevo. Con esto, con el mismo botón, podemos realizar ambas labores, el llenado y vaciado.

- **TSpeedButton**: Para poder manipular el control de las consignas de temperatura de la cámara, se ha utilizado 2 botones, uno para bajar y otro para subir las consignas de marcha y paro de la misma. Dado que la operación a realizar es la misma, salvo que el valor se incrementa o disminuye en 1ºC, ambos botones utilizan el mismo evento "OnClick", y dentro del mismo, mirando el nombre del objeto que lo llama, sabemos si el valor debe de ser +1 o -1. También se tendrá en cuenta, si estamos viendo la consigna de marcha o de paro, para modificar la seleccionada. Para evitar que se seleccionen valores que estén fuera del rango de trabajo, se ha controlado que no se puedan superar los 40ºC, temperatura máxima fijada como temperatura ambiente. En el caso de la consigna de paro no podrá fijarse un valor por debajo de -3ºC, y la consigna de marcha no podrá ser inferior o igual a la de paro. Cada vez que se varía el valor, se actualiza el nuevo valor en el componente TStaticText que muestra la temperatura "STCamara", y por supuesto, el nuevo valor se envía a la tarjeta virtual.

- **TUpDown**: Mediante este objeto, dentro del control de temperatura de la cámara, podremos cambiar el tipo de temperatura que estamos visualizando, temperatura actual de la cámara, consigna de marcha o consigna de paro de la cámara. Cada vez que se pulsa este objeto, se actualiza la imagen cargada en el componente "ImgTMP" y se modifica el valor de la variable "tipo_temperatura", para saber en todo momento, que temperatura es la que estamos mostrando. También controlamos que al estar visualizando la temperatura de la cámara, los botones para subir o bajar grados del control, no estén activos, puesto que sólo lo deben de estar para cambiar consignas.

- **TPanel**: Este control, sirve para agrupar otros objetos de la aplicación, es un mero contenedor de objetos. En este caso lo utilizamos para agrupar los elementos que forman el control de temperatura de la cámara ("PControlCamara"), así como en el caso de los objetos que forman el control de temperatura ambiente ("PTempAmbiente").

- **TTrackBar**: Se ha utilizado este componente, para que el usuario pueda simular la variación de temperatura ambiente. Se han utilizado sus propiedades "Min" y "Max" para limitar el rango de valores que se pueden mostrar, en este caso entre 0 y 40, respectivamente. Se ha utilizado su evento "OnChange", para controlar que cada vez que se deslice la barra, se actualice el valor de la propiedad "Caption" del TLabel "LTempAmbiente", y por supuesto se escriba en la tarjeta virtual, el valor en cuestión.

5.4 Historial de versiones

En el transcurso del presente proyecto, se han ido realizando diferentes versiones de la aplicación que satisfacían los requisitos implementados. Incluiremos la descripción del progreso de estas versiones:

❖ **Versión 28 de junio de 2012**

- Basándome en el código de ejemplo suministrado por el director del proyecto, consigo realizar la adaptación de este al entorno de desarrollo de Borland Delphi 7. Se traduce el código escrito en lenguaje C++ al lenguaje Pascal, se definen las variables necesarias, etc.
- La aplicación es capaz de comunicar con la tarjeta virtual.

❖ **Versión 3 de julio de 2012**

- Primer boceto en CorelDraw X4 del diseño con la distribución de la planta representada. Ya se incluyen los dibujos del cuadro eléctrico, compresor, condensador, separador de amoniaco, depósito de glicol y la cámara frigorífica, así como la mayoría de las tuberías que los unen.
- Se preparan las imágenes que representarán los distintos estados: ventiladores en varias posiciones para representar el movimiento, bombas y motor en varios colores y puerta abierta y cerrada.
- Añadidos los distintos procedimientos encargados de la simulación del estado del motor, ventiladores del condensador, bomba del condensador y bombas de primario y secundario.
- Se incluye el control para abrir y cerrar la puerta de la cámara.
- Añadidos los eventos "Onclick" en la imagen del interruptor general y de los diferencias del cuadro. Ya cambian de imagen en función de su estado y envían la correspondiente señal a la tarjeta virtual.

❖ **Versión 8 de julio de 2012**

- Dibujados los ventiladores de la cámara y la sirena de hombre encerrado, con sus distintos colores. En la aplicación se incluyen los procedimientos encargados de simular sus respectivos movimientos.
- Pruebas de escritura de las analógicas, utilizando botones y comprobando si los elementos se activan o no, en función de las distintas temperaturas.
- Pruebas con una barra de deslizamiento para controlar la temperatura ambiente.

❖ **Versión 11 de julio de 2012**

- Diseñado el control de temperatura de la cámara. Realizadas las imágenes que identificarán la temperatura que se está mostrando en el visualizador. Programados los eventos de los botones, encargados de cambiar en el visualizador el tipo de temperatura mostrada. Programados los eventos de los botones, encargados de modificar las consignas de marcha y paro del control.
- Dibujada la imagen para el control de la temperatura ambiente y diseñado e implementado este control.

- **Versión 29 de julio de 2012**

  - Diseñado e implementado el algoritmo para simular la variación de temperatura en la cámara según las condiciones de temperatura ambiente, puerta abierta, cámara llena, ...
  - Diseñado e implementado el algoritmo para simular la variación de temperatura del depósito de glicol según las condiciones de temperatura ambiente y compresor en marcha.

- **Versión 27 de agosto de 2012**

  - Se incluye en el cuadro eléctrico el selector para el habilitado de la cámara. Se incluye esta señal en el programa de C.
  - Se acaban de dibujar las tuberías que faltaban así como la unión con sus respectivos elementos. Se cambian de posición algunos elementos para que quede más limpio el cruce de tuberías.
  - Dibujada la imagen para representar el género en la cámara.
  - Programado el evento “OnClick” del botón encargado del llenado y vaciado de la cámara.
  - Depuración de código fuente.

- **Versión 10 de septiembre de 2012**

  - Solucionado problema en el algoritmo que simula la variación de temperatura del glicol.
  - Añadidas mejoras en la interfaz de usuario propuestas por el director del proyecto, agrandado el control de temperatura de la cámara para facilitar su manejo y su visualización.
  - Solucionado problema con el control de temperatura de la cámara, detectado por el director del proyecto. Se reemplaza el componente “TUpDown” encargado de modificar las consignas de marcha y paro, por unos botones “TSpeedButton”. Al utilizarse el mismo componente “TUpDown” tanto para el control de la consigna de paro como el de la consigna de marcha, hacía que se produjeran problemas al cambiar de control.
  - Corregido error en la temperatura de la cámara mostrada, cuando se llena o se vacía la cámara.
  - Depuración de código fuente

- **Versión final**

  - Limpieza final de código. Eliminación de todo el código comentado por pruebas. Eliminación de todas las variables no utilizadas.
  - Versión final de la aplicación satisfaciendo todos los requisitos.

6 PRUEBAS

Para probar el correcto funcionamiento del simulador gráfico, y que todos los elementos respondían adecuadamente ante los cambios que se realizaban en este, se ha realizado un completo programa en C, que representa lo que sería en una instalación real, un programa de un autómata, el cual se encarga de realizar el control de la planta. Este programa será un ejemplo del que tendrán que hacer los alumnos en el futuro, cuando realicen sus prácticas.

Aquí se muestra el código de dicho programa, con sus correspondientes comentarios que facilitan la comprensión del mismo:

```
int main(int argc, char* argv[])
{
 short card;
 short err;
 short salida;
 unsigned long digi;
 unsigned long analog;
 unsigned long salida_digital;
 unsigned long analogica;

 float consigna_paro;
 float consigna_marcha;
 float temperatura_camara;
 float temperatura_glicol;

 char interruptor_general = '0';
 char dif_compresor = '0';
 char dif_ventilador_condensador = '0';
 char dif_bomba_condensador = '0';
 char dif_bomba_primario = '0';
 char dif_bomba_secundario = '0';
 char dif_ventilador_camara = '0';
 char puerta_abierta = '0';
 int marcha_camara = '0';
 int hombre_encerrado = 0;
 int marcha_primario = 0;
 int marcha_secundario = 0;
 int enfriar;
 int enfriar_glicol;

 printf ("\nHola\n");
 card = Register_Card(PCI_9112, 0);
 if (card<0)
 {
        printf ("\error %d\n",card);
```

```
}

printf ("\n Este programa controla el proceso de frío industrial");
printf ("\n ******** PULSA UNA TECLA PARA CONTINUAR ************");
getch();

while (!kbhit())
{
    err = DI_ReadPort (card, 0, &digi);
    interruptor_general = (digi & 0x0001);
    dif_compresor = (digi & 0x0002);
    dif_ventilador_condensador = (digi & 0x0004);
    dif_bomba_condensador = (digi & 0x0008);
    dif_bomba_primario = (digi & 0x0010);
    dif_bomba_secundario = (digi & 0x0020);
    dif_ventilador_camara = (digi & 0x0040);
    puerta_abierta = (digi & 0x0080);
    if (digi & 0x0100)
      hombre_encerrado = 1;
    else hombre_encerrado = 0;
    if (digi & 0x0200)
      marcha_camara = 1;
    else marcha_camara = 0;

    salida_digital = 0;
    // Si no está activo el interruptor general no se hace nada, ya que la instalación no tiene corriente
    if (interruptor_general)
    {
        err = AI_ReadChannel (card, 0, 2, &analog);
        analog = analog >> 4;
        consigna_paro = (100.0*analog/4095)-40;

        err = AI_ReadChannel (card, 1, 2, &analog);
        analog = analog >> 4;
        consigna_marcha = (100.0*analog/4095)-40;

        err = AI_ReadChannel (card, 2, 2, &analog);
        analog = analog >> 4;
          temperatura_camara = (100.0*analog/4095)-40;

        err = AI_ReadChannel (card, 3, 2, &analog);
        analog = analog >> 4;
          temperatura_glicol = (100.0*analog/4095)-40;

    // Controlo si hay que hacer frío, teniendo en cuenta si la temperatura de la cámara está por
    // encima de la consigna de marcha indicada o parándolo cuando se alcanza la consigna de paro,
    // y siempre que esté habilitada la cámara desde el cuadro
    // Controlo si hay que enfriar el glicol, teniendo en cuenta si la cámara está habilitada y
```

```
// teniendo en cuenta que para enfriar a determinada temperatura, el glicol debe de estar como
// mucho 8º C por debajo de esta

    if (marcha_camara)
    {
            if (temperatura_camara >= (consigna_marcha))
                    enfriar = 1;
            else
            if (temperatura_camara <= (consigna_paro))
                    enfriar = 0;

            if (temperatura_glicol >= (consigna_marcha - 8))
                    enfriar_glicol = 1;
            else
            if (temperatura_glicol <= (consigna_paro - 8))
                    enfriar_glicol = 0;
    }
    else
    {
            enfriar = 0;
            enfriar_glicol = 0;
    }

// Si se para el compresor, la bomba del condensador o la bomba del primario,
// se para el circuito primario, y se marca la alarma en cuestion. Si se para
// el ventilador del condensador, mientras funcione la bomba no se para el circuito

    if (enfriar_glicol && dif_compresor && dif_bomba_condensador && dif_bomba_primario)
            marcha_primario = 1;
    else marcha_primario = 0;

// Si están en marcha la bomba del secundario y el ventilador de la cámara, y la temperatura del
// glicol no está por encima de la consigna de marcha - 8º, se mantiene en marcha el secundario

    if (enfriar && dif_bomba_secundario && dif_ventilador_camara &&
      (temperatura_glicol < (consigna_marcha - 8)))
            marcha_secundario = 1;
    else marcha_secundario = 0;

// Controlo los estados de las salidas digitales
    if (dif_compresor)
    {
            if (marcha_primario)
                    salida_digital = salida_digital + 1; //marcha compresor
    }
    else salida_digital = salida_digital + 128; //avería eléctrica compresor

    if (dif_ventilador_condensador)
```

```
        {
                if (marcha_primario)
                        salida_digital = salida_digital + 2; //marcha ventilador condensador
        }
        else salida_digital = salida_digital + 256; //avería eléctrica ventilador condensador

        if (dif_bomba_condensador)
        {
                if (marcha_primario)
                        salida_digital = salida_digital + 4; //marcha bomba condensador
        }
        else salida_digital = salida_digital + 512; //avería eléctrica bomba condensador

        if (dif_bomba_primario)
        {
                if (marcha_primario)
                        salida_digital = salida_digital + 8; //marcha bomba primario
        }
        else salida_digital = salida_digital + 1024; //avería eléctrica bomba primario

        if (dif_bomba_secundario)
        {
                if (marcha_secundario)
                        salida_digital = salida_digital + 16; //marcha bomba secundario
          }
        else salida_digital = salida_digital + 2048; //avería eléctrica bomba secundario

        if (dif_ventilador_camara)
        {
                if (marcha_secundario)
                        salida_digital = salida_digital + 32; //marcha ventiladores cámara
        }
        else salida_digital = salida_digital + 4096; //avería eléctrica ventiladores cámara

    // Si la puerta está cerrada y se pulsa el botón de hombre atrapado, hago sonar la sirena
        if (!puerta_abierta && hombre_encerrado)
                salida_digital = salida_digital + 64; //sirena hombre atrapado
      }

      err = DO_WritePort (card, 0, salida_digital);

      Sleep (100);
 }
    getch();
    return 0;
}
```

Para completar las pruebas en la aplicación gráfica, se han ido simulando las distintas situaciones que se pueden dar en una instalación de este tipo. Se han provocado averías en los elementos del circuito de frío, comprobando que el resto de elementos respondían de la forma esperada.

En el caso de las temperaturas, se ha comprobado que la temperatura del glicol varía en función de las distintas condiciones ambientales, así como de que esté en marcha el circuito primario.

Para las pruebas en la cámara se han simulado diferentes estados en esta, con y sin género, con la puerta abierta y cerrada, con el circuito primario o secundario parados por alguna avería, y por supuesto teniendo en cuenta en todo momento la temperatura ambiente, la cual afecta de forma directa a la velocidad con la que la cámara se enfría o calienta.

Después de corregir los errores detectados, podemos concluir que se han pasado todos los test realizados y que tenemos una aplicación robusta y apta para cumplir con su cometido.

7 FUTURAS IMPLEMENTACIONES

Entre las posibles ampliaciones que se podrían realizar al presente proyecto podría estar el incluir más animaciones en los elementos. Por ejemplo, se podría hacer que las tuberías simularan el movimiento de los líquidos.

También podría simularse distintos estados de carga de género en la cámara, para refinar aún más el algoritmo que simula la variación de temperatura de la misma.

Por otro lado, para el tema de las averías en los elementos de la instalación, podría programarse que de forma aleatoria se produjeran fallos en estos, y que esto disparara el diferencial del cuadro, en lugar de simularlo pulsando directamente en este, como se está haciendo ahora.

Incluso podría incluirse fugas de gases o incendios en la planta, que forzaran a repararlo de alguna forma para que todo volviera a la normalidad.

En el controlador de temperatura del glicol, también podría incluirse un control similar al de la cámara, en el cual se pueda programar manualmente las consignas de marcha y paro, en lugar de hacerlo en función de lo programado en la cámara.

Otra posible ampliación, que daría más realismo al simulador, sería el incluir más cámaras en la planta. De esta forma, se podría ver más claramente que ocurre en el circuito cuando hay demanda de frío de varias cámaras, tendría más sentido el poder habilitar o no desde el cuadro algunas de ellas, etc.

8 CONCLUSIONES

Cumpliendo con los requisitos propuestos, se ha realizado una aplicación que pondrá en manos de los alumnos una completa herramienta gráfica, que representa con gran exactitud el funcionamiento real de una instalación de frío industrial.

Con ella podrán realizar sus prácticas tanto en los laboratorios como en sus propias casas, simulando las situaciones habituales que se dan en este tipo de instalación, sin tener que desplazarse a estas, y por supuesto sin tener que disponer de estos equipos, compresores, condensadores, etc., a los cuales obviamente no se tiene acceso salvo que trabajes para alguna empresa que se mueva en este sector.

Se ha realizado el correspondiente control de versiones, documentando las mismas.

La comunicación con el director ha sido lo suficientemente clara, para saber en todo momento el alcance del presente proyecto, y conseguir que se este se haya podido llevar a cabo en los tiempos previstos.

En resumen, se ha alcanzado el objetivo fijado, disponiendo de una herramienta fiel al proceso que simboliza, un proceso de frío industrial.

9 AGRADECIMIENTOS

A los compañeros del trabajo que me han ayudado a comprender el funcionamiento real de una instalación de frío industrial, en especial a José Pascual Ferrer Ferrer y Francisco Calatayud Sampedro, que han aportado su dilatada experiencia, como programadores de los autómatas empleados en este tipo de instalaciones.

Por supuesto, al director del proyecto Antonio Martí Campoy, por haber comprendido mi situación personal y laboral. Por otro lado, agradecer el interés mostrado en todo momento en el avance del proyecto, inclusive durante su periodo de vacaciones, y por toda la ayuda prestada, para comprender el funcionamiento de la tarjeta virtual y la librería, que debía utilizar para realizar este proyecto.

Tampoco quiero olvidarme de mis padres, los cuales me han apoyado en todo momento para que finalizara la carrera, a pesar de que esto se haya prolongado mucho en el tiempo.

Por último, agradecer a mi mujer y mis hijos, por todo ese tiempo, tardes, fines de semana, vacaciones... en los cuales no les he podido prestar la atención que requerían, por tener que dedicarlo a mis estudios y en último lugar a este proyecto.

10 BIBLIOGRAFÍA

- Memoria del proyecto final de carrera de Miguel Carro Pellicer: "Simulador De Tarjeta De Adquisición De Datos Nudaq/Nuipc 9112 Series".17/07/07 (Director Académico: Martí Campoy, Antonio)
- MSDN library online, consulta de sintaxis de las llamadas del lenguaje C.
- Biblioteca de dibujos de elementos de instalaciones de frío, desarrollados en la empresa Diselcom S.L. (empresa de la cual soy responsable del Departamento de Informática Interna.)
- Delphi Help. Consulta de propiedades, eventos, funciones, etc. de los objetos de Delphi.
- Microsoft Windows Setup API Programer's Reference. Consulta de la estructura en las funciones API de Windows.

11 ANEXO 1: MANUAL DE USUARIO

11.1 Introducción

Se ha planteado una instalación de frío industrial, la cual utiliza amoniaco, como gas para conseguir el frío. Consideraremos que sólo existe una cámara y en esta sólo es necesario alcanzar una temperatura de -3º C. Esta consideración es importante, puesto que si necesitáramos alcanzar temperaturas inferiores, no podríamos utilizar agua glicolada, y habría que llevar directamente el gas a la cámara.

El utilizar el agua, abarata la instalación puesto que es mucho más económica que el gas. Podemos suponer que la cámara se encuentra a varios cientos de metros de la sala de máquinas, y si lleváramos el gas hasta la cámara, sería necesario una gran cantidad de este, para llenar todo el circuito. A todo esto, hay que añadir la ventaja que supone trabajar con agua, la cual ante una fuga, no impide que se pueda seguir trabajando en la cámara, y por supuesto, no afectaría al género allí almacenado.

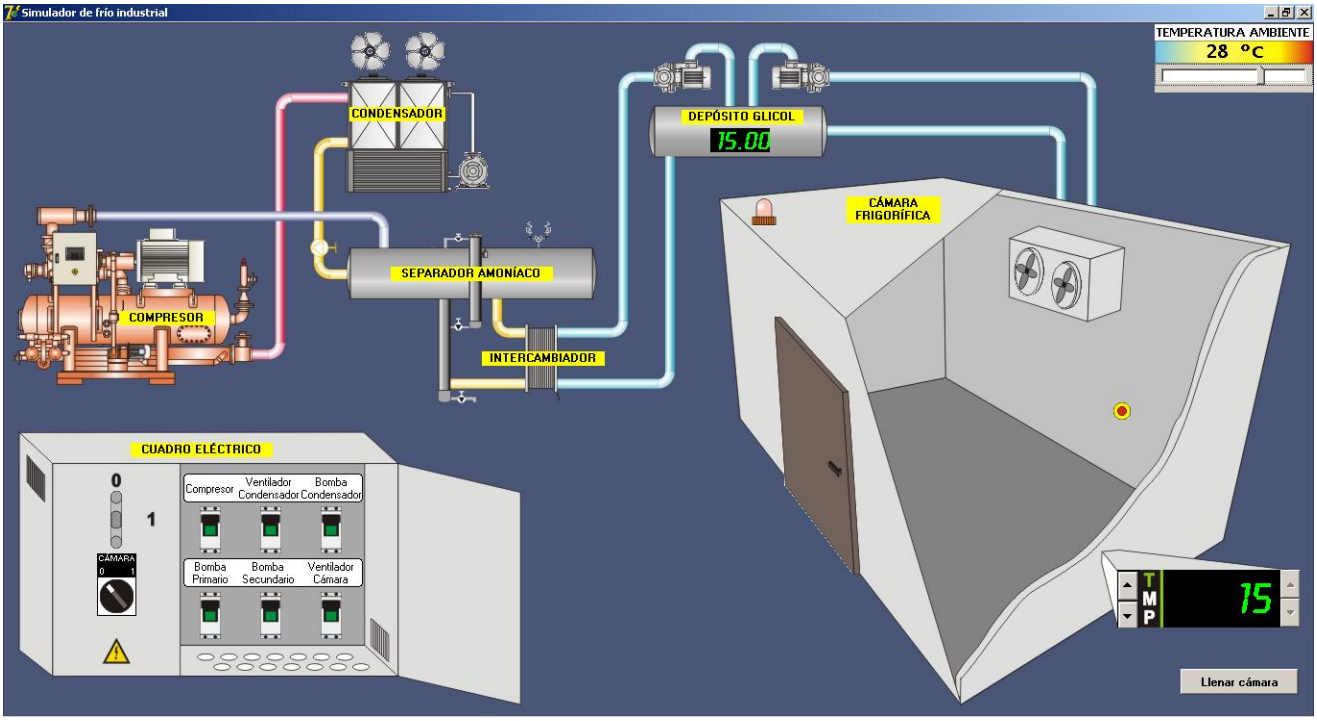
En esta instalación, vamos a encontrar 2 circuitos.

a. El primario está constituido por los siguientes elementos:
    i. Compresor: Encargado de comprimir el gas utilizado para obtener el frío, en este caso amoniaco.
    ii. Condensador: Encargado de condensar el gas proveniente del compresor, y convertirlo en líquido. Para ello emplea tanto los ventiladores como una bomba de agua.
    iii. Separador: Recipiente en el cual, mediante una válvula de expansión, el refrigerante líquido, proveniente del condensador, se transforma en gas, consiguiendo una temperatura muy baja.
    iv. Intercambiador: Elemento formado por 2 tuberías por las cuales se hace pasar el amoniaco muy frío proveniente del separador y por la otra agua glicolada, consiguiendo que esta última alcance la temperatura necesaria para posteriormente enfriar la cámara en cuestión.
    v. Bomba de glicol: Bomba encargada de hacer circular el agua glicolada entre el recipiente de glicol y el intercambiador.
    vi. Recipiente de glicol (agua glicolada): Elemento que sirve de unión entre ambos circuitos, en el cual se almacena el agua glicolada.

b. El secundario formado por los elementos:
    i. Recipiente de glicol (agua glicolada): Elemento que sirve de unión entre ambos circuitos, en el cual se almacena el agua glicolada.
    ii. Bomba de glicol: Bomba encargada de hacer circular el agua glicolada entre el recipiente de glicol y la cámara.
    iii. Ventiladores de la cámara: Encargados de impulsar aire a la cámara. Este aire se hace pasar a través del circuito de tuberías con el agua glicolada fría y consiguiendo que entre frío en la cámara.

11.2 Manejo de la aplicación

A continuación vamos a detallar como podemos interactuar con los diferentes elementos de la instalación, la cual presenta el siguiente aspecto.

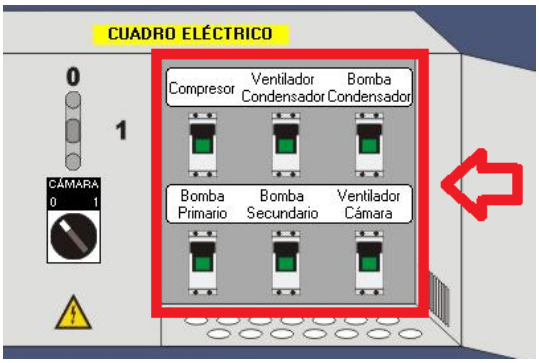


Para empezar, para que se ponga en marcha el circuito, lo primero que se debe hacer es conectar el interruptor general del cuadro eléctrico, ya que sin corriente no funciona ningún elemento.

También será necesario que se habilite la cámara, ya que si no se precisa enfriarla, no es necesario hacer frío. Con este botón de habilitado, se controla que no se malgaste energía enfriando una cámara en la cual no se prevé que se vaya a introducir género en breve. Se puede pensar que este botón es innecesario, puesto que para controlar esto se podría simplemente no activar el interruptor general del cuadro. Es este proceso simple que se propone es así, pero si pensamos en una instalación real, en la cual podrían haber varias

decenas de cámaras distintas, este botón es necesario puesto que es bastante habitual que no se estén utilizando todas las cámaras, en todo momento.

Para simular averías eléctricas en alguno de los elementos de la instalación, simplemente pulsaremos sobre los diferenciales del cuadro eléctrico para simular que ha saltado dicho diferencial.



La misión del circuito primario es que el depósito de glicol tenga la temperatura necesaria para enfriar la cámara. Estará en marcha, mientras no haya una avería en alguno de sus elementos, no se alcance la temperatura del glicol necesaria para enfriar adecuadamente la cámara y la cámara esté en marcha. En el caso de que se produzca una avería en el ventilador del condensador mientras funcione la bomba es posible condensar, por lo que el circuito seguiría en marcha, aunque el ventilador se parará hasta que se solucione el problema.

El circuito secundario es el encargado de conseguir que la cámara mantenga la temperatura deseada. Estará en marcha, siempre que la cámara esté en funcionamiento, su temperatura se encuentre por encima de la consigna de paro, no haya una avería en alguno de sus elementos y la temperatura del glicol sea suficiente para conseguir la temperatura deseada.

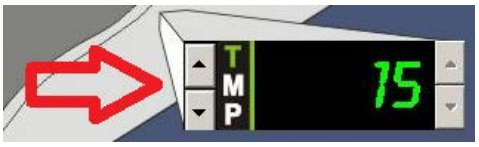
Para simular la variación de la temperatura de la cámara, se ha tenido en cuenta si el ventilador está en marcha y la temperatura ambiente, dividiendo esta en franjas de 0°C a 9°C, de 10°C a 19°C, de 20°C a 29°C y de 30°C a 40°C.

También se tendrá en cuenta si la puerta está abierta o cerrada y si la cámara está llena o vacía. Cuesta más enfriar género que el aire de una cámara vacía.

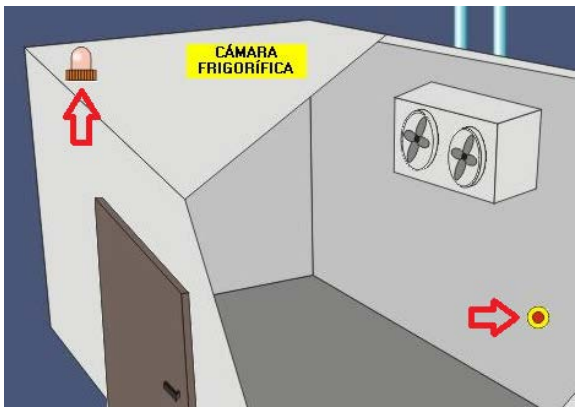
Para simular la variación de la temperatura del glicol, se ha tenido en cuenta si está en marcha el compresor (con controlar un elemento es suficiente, puesto que si falla alguno se paran todos) y la temperatura ambiente, dividida en las mismas franjas que para la temperatura de la cámara. A mayor temperatura ambiente, más lentamente se alcanza la temperatura deseada.

Para alcanzar una determinada temperatura en la cámara, el glicol debe de estar 8º C por debajo de esa temperatura. Existen distintas formas de configurar esto. Una podría ser incluir una consigna de marcha y otra de paro, en el depósito de glicol, pero para simplificar, utilizaremos las mismas de la cámara teniendo en cuenta que para el glicol serán 8º C menos. Por ejemplo, si configuramos en la cámara una consigna de marcha de 5º C y una consigna de paro de 0º C, para el glicol utilizaremos una consigna de marcha de -3º C y una de paro de -8º C.

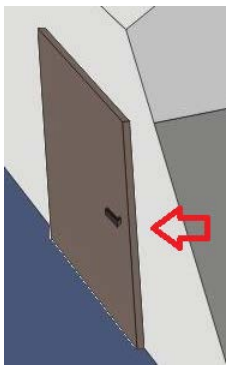
La cámara dispone de un controlador, en el cual se puede visualiza la temperatura actual, así como configurar las consignas de marcha y paro. Utilizaremos las flechas de la izquierda para cambiar la temperatura que estamos visualizando, temperatura actual, temperatura de marcha o temperatura de paro. Utilizaremos las flechas de la derecha, para subir o bajar en un grado, la temperatura de marcha o paro, según la que se esté visualizando.



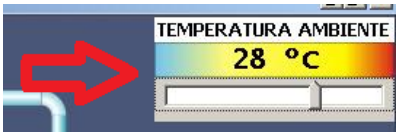
En el interior de la cámara, existe una seta de emergencia para avisar de que hay un hombre atrapado. Si estando la puerta cerrada se pulsa esta, se activará la sirena que avisa de esto.



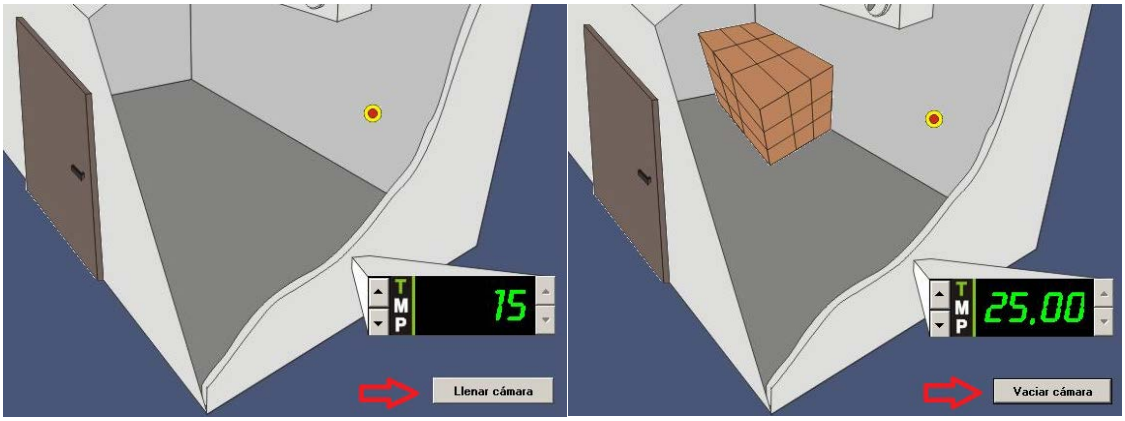
Para abrir o cerrar la cámara, simplemente se pulsará sobre la puerta de esta.



En la parte superior derecha de la pantalla, disponemos de un control para simular los cambios de temperatura ambiente. Para realizar esto, solo tendremos que utilizar la barra de deslizamiento.



Para acabar, en la parte inferior derecha, existe un botón para simular el llenado y vaciado de la cámara con un género indeterminado. Simplemente pulsaremos sobre este para alternar entre llena o vacía.



11.3 Tabla de entradas y salidas

A la hora de realizar el programa de prácticas, es absolutamente necesario saber la correspondencia entre cada uno de los elementos de la aplicación que interactúan con la tarjeta virtual y bit o canal de la tarjeta. Esta relación la mostramos a continuación, indicando además su rango o el significado para cada uno de los estados, en el caso de las entradas y salidas digitales.

Entradas digitales		
Bit	Señal	Estados
0	Interruptor general del cuadro eléctrico	1 - Conectado / 0 - Desconectado
1	Diferencial del compresor	1 - Ok / 0 - Avería
2	Diferencial del ventilador del condensador	1 - Ok / 0 - Avería
3	Diferencial de la bomba del condensador	1 - Ok / 0 - Avería
4	Diferencial de la bomba del circuito primario	1 - Ok / 0 - Avería
5	Diferencial de la bomba del circuito secundario	1 - Ok / 0 - Avería
6	Diferencial del ventilador de la cámara	1 - Ok / 0 - Avería
7	Sensor de puerta abierta	1 - Abierta / 0 - Cerrada
8	Seta de emergencia de hombre encerrado en cámara	1 - Activada / 0 - No activada
9	Paro/marcha de la cámara	1 - Marcha / 0 - Paro

Salidas digitales		
Bit	Señal	Estados
0	Marcha compresor	1 - Marcha / 0 - Parado
1	Marcha ventilador del condensador	1 - Marcha / 0 - Parado
2	Marcha bomba del condensador	1 - Marcha / 0 - Parado
3	Marcha bomba del circuito primario	1 - Marcha / 0 - Parado
4	Marcha bomba del circuito secundario	1 - Marcha / 0 - Parado
5	Marcha ventiladores de la cámara	1 - Marcha / 0 - Parado

Entradas analógicas		
Canal	Señal	Rango
0	Consigna de paro de la cámara	-3ºC .. 40ºC
1	Consigna de marcha de la cámara	-2ºC .. 40ºC
2	Temperatura de la cámara	-5,75ºC .. 40ºC
3	Temperatura del depósito de glicol	-12,75ºC .. 40ºC

Hay que tener en cuenta que se ha simulado la lectura de temperatura utilizando unas sondas de temperatura que leen valores entre -40ºC y 60ºC y que habrá que realizar el correspondiente escalado.

En el programa de prácticas, dado un valor de una entrada analógica “v_analogica”, devuelto por la función “AI_ReadChannel”, este deberá ser escalado de la siguiente manera:

```
v_analogica = v_analogica >> 4;
valor_escalado = (100.0 * v_analogica/4095)-40;
```