Capitulo 2

Ha saltado la alarma.

Crear alarmas.

Llegamos a nuestro capítulo preferido: generar alarmas. Lo primero que hay que tener claro es qué deseamos mostrar en la instalación:

- Alarmas de las que acojonan, y para que se quiten hay que tocar un botón.
- Avisos de esos que nadie lee, y que si desaparecen del plc nadie se entera.
- Otras cosas, que no son nada de esto, y que utilizamos las alarmas para mostrarlas, p. Ej. Un histórico de usuarios, para saber cuando se ha logado un operario y por cuanto tiempo.

Teniendo claro todo esto, vamos a por las alarmas normales, las de toda la vida. Tenemos tres posibilidades de ver/guardar las alarmas: las actuales, un histórico en formato tambor[1], y todas las que vengan. Yo no recomiendo esta última opción, ya que el WinCC no tiene manera de quitar fácilmente este histórico de la base de datos, , por lo que nos quedan la de actuales y la de tambor.

Se suele gastar la visualización de las actuales en la picture bottom.pdl, poniendo una ventanita pequeña que nos muestra la última alarma que se está produciendo en la instalación, y se suele poner un botón de reset de alarma, para no tener que ir a la ventana de alarmas si es una tontería, y poder resetearla sin cambiar de pantalla. Si la cosa pinta fea,

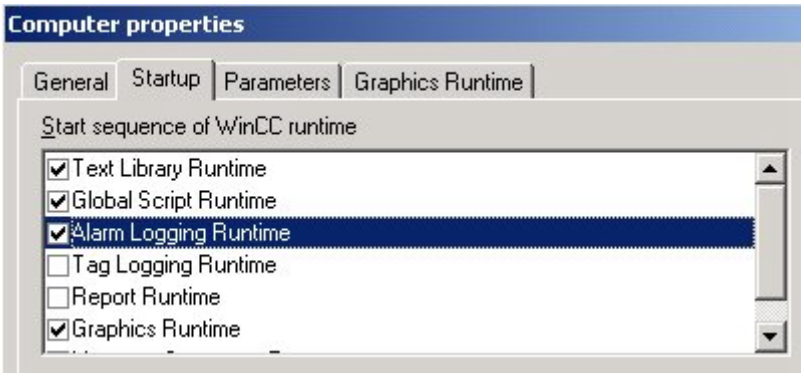
Y la alarma puede tener tela, se suele poner un botón de salto a la ventana de alarmas, en la que se muestra el histórico de alarmas en formato tambor, y ya se incluyen en esta ventana todas las opciones posibles en tratamientos de alarmas del WinCC, como filtrados, bloqueos, impresión, etc..

Alarmas actuales.

Bien, nuestro objetivo va a ser visualizar la alarma que se produzca en la instalación, estando en cualquier ventana del proyecto, y poder resetearla.

Antes de nada, vete a computer->tu_ordenador (sieneo tu_ordenador el nombre de tu ordenador), y en la solapa startup, seleccionar el runtime de alarm logging, como muestra la siguiente figura. Si no se hace esto, luego se olvida uno y cuando va a comprobar como funcionan sus alarmas, salen menos que los monjes de clausura de fiesta. Así que actívalo ya y así seguro que no se te olvida.

[1] Tambor: se dice tambor en programación a ir almacenando valores hasta un valor límite, a partir del cual el primer valor introducido se pierde a cambio del siguiente que se va a almacenar, por lo que siempre tendremos los últimos x valores en el buffer.

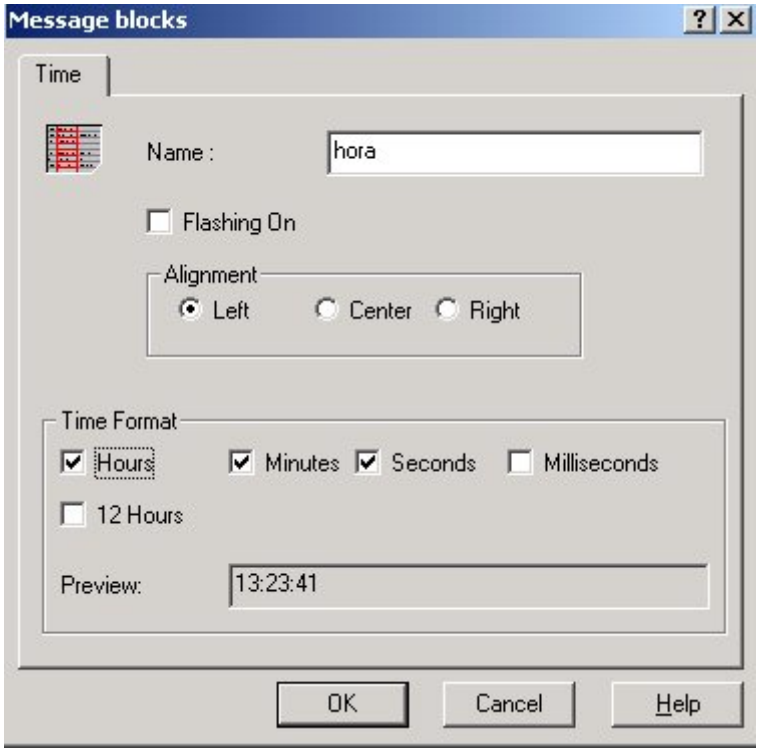


Abrimos el Alarm Logging en el control center de WinCC y comenzamos a editar los diferentes bloques de los cuales se va a componer nuestra línea de alarmas. Vamos a utilizar:

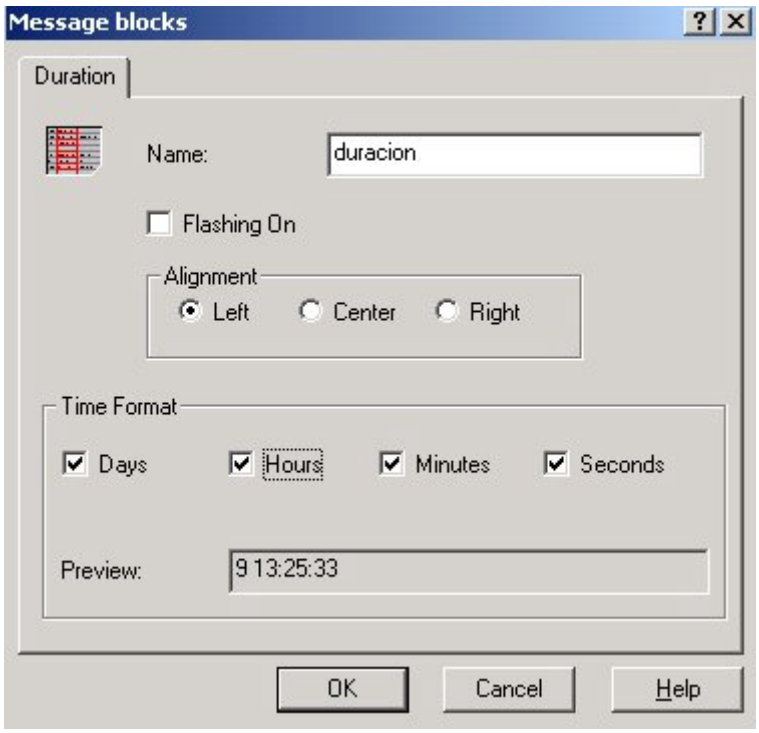
- La fecha
- La hora
- La duración
- El número de alarmas
- El estado actual (activa, desactiva, reconocida, desactiva sin haber sido reconocida).

Lo primero añadiremos el de duración, que no aparece por defecto. En message blocks->system blocks añadimos duration y status.

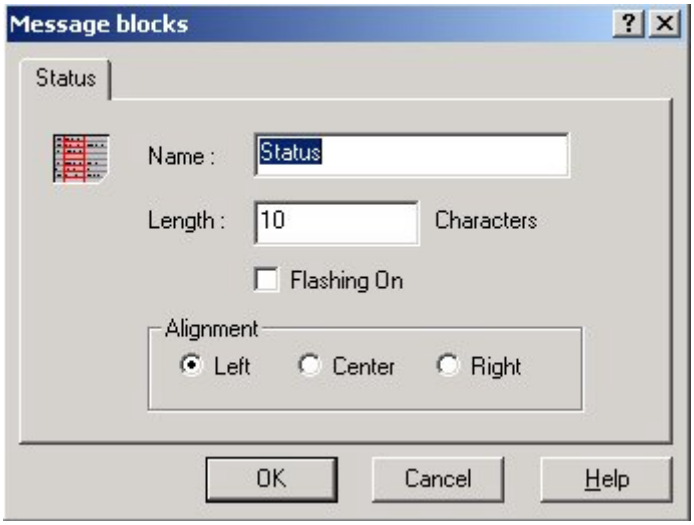
Comenzamos a modificar los bloques, para que aparezcan en castellano los titulos y algunas cosas mas. Mostraré en concreto el de la hora:



La duración de la alarma dentro del sistema:

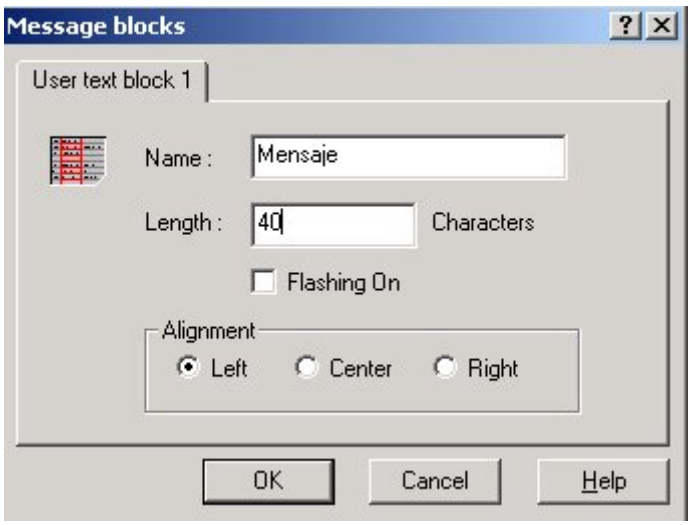


El status actual de la alarma. Hay que poner un 10 en longitud (length) porque si no luego no hay manera de que aparezca aquí un texto decente que no sea del estilo +/#"@, que es lo que el wincc pone, a modo de dialogo de mortadelo y filemon insultando.

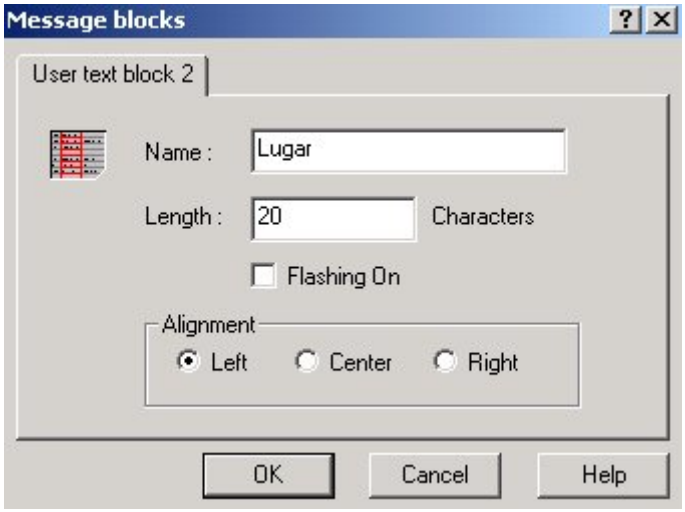


Luego vamos a poner aquí unos textos del tipo: aparecede, desaparece, etc..
Al final quedan mas o menos estos bloques de sistema:

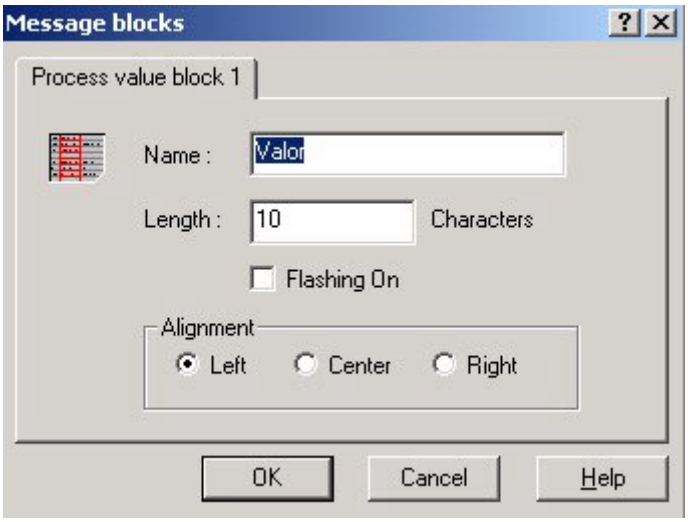
Vamos ahora con los mensajes de usuario, que los dos que nos ponen están bien, salvo que debemos de cambiarles la longitud, porque si no a ver como metemos ahí un texto. Comenzamos con el message text, que queda así:



Y el de lugar donde se produce la alarma, nos queda así:

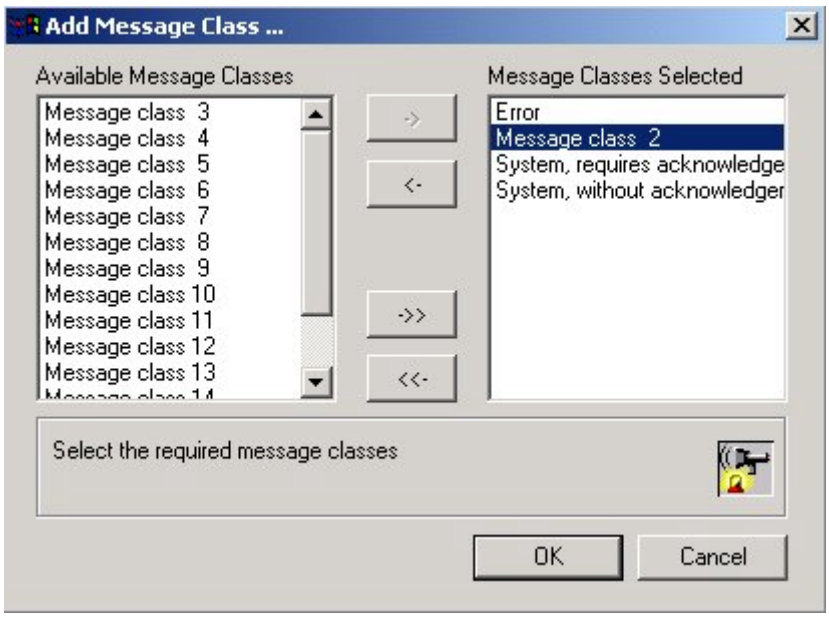


Por último, vamos a poner un bloque de proceso, por si necesitamos mostrar un valor de proceso en la línea de alarmas.

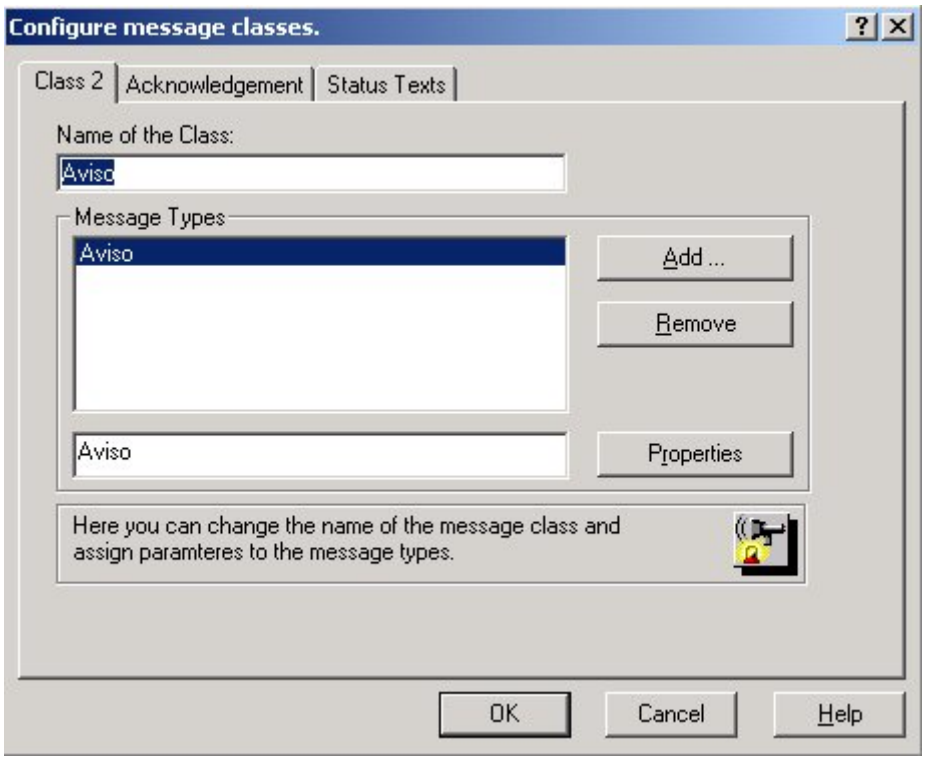


Bien, ya teniendo definidos los bloques que van a componer nuestra línea de alarmas, pasamos a message classes. Aquí definimos los tipos o clases de alarmas o mensajes que vamos a realizar. En nuestra instalación vamos a tener alarmas, que se quedan y hacen pupita, y es necesario acusarlas, y avisos, que si lo miras bien y si no también. El sistema nos da tres clases predefinidas: Error, que es donde vamos a poner nuestras alarmas, system with acknowledgement, o sea, alarmas generadas por el sistema y que deben ser reconocidas, y system without acknowledgement, avisos del sistema.

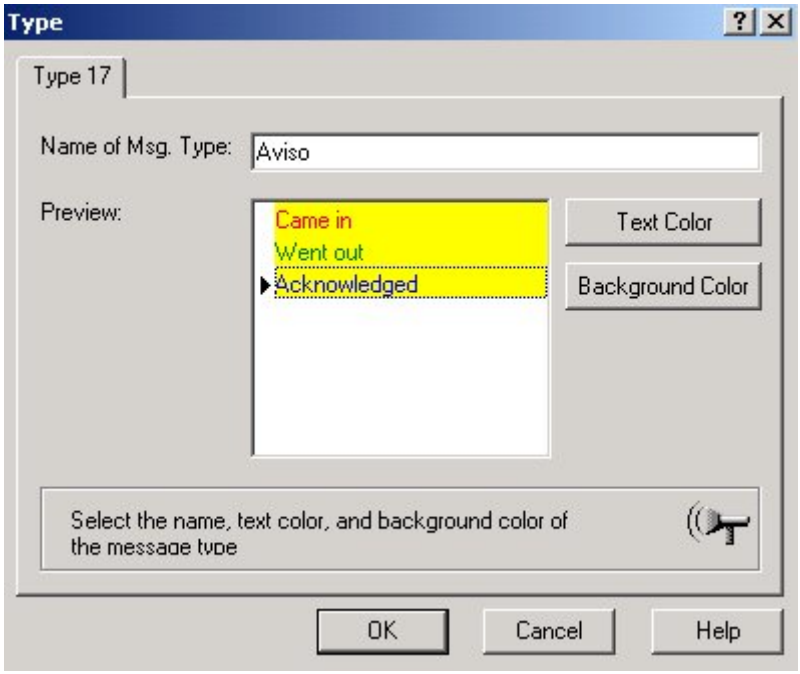
Como tenemos avisos, y no deben ser reconocidos, nos creamos una nueva clase que sea aviso



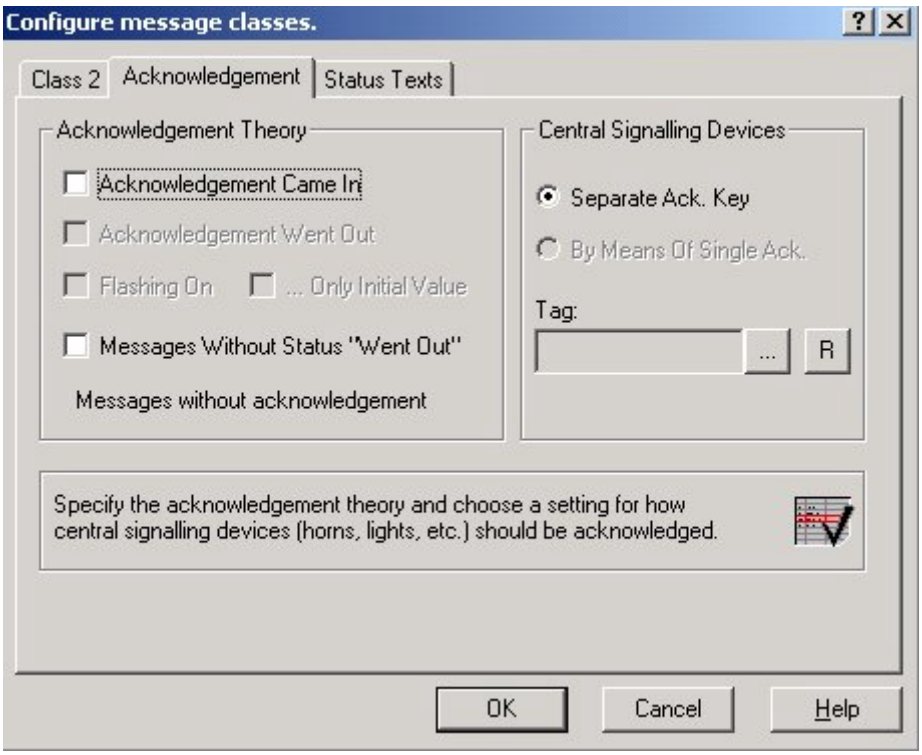
Aunque aparece lo de message class 17, luego se lo cambiamos por Aviso y a correr. Creamos un type dentro de esta clase que sea aviso, como muestra la siguiente figura.



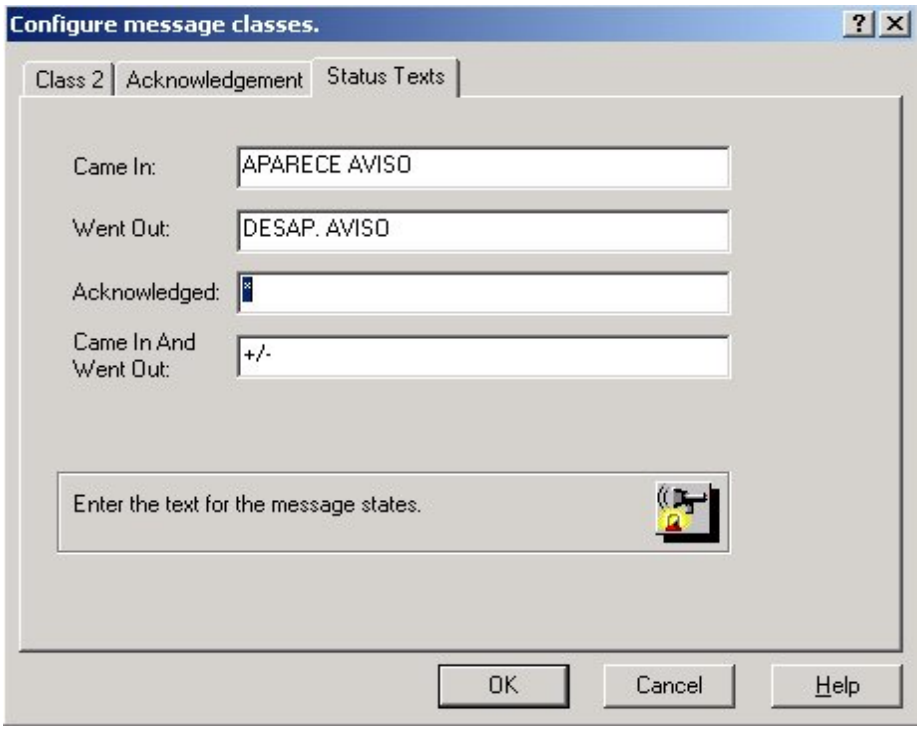
Este type aviso le damos color de fondo amarillo para cualquier situación: aparece, desaparece, reconocida. Distinguiremos si es un caso u otro a través del color de la fuente del texto, que cuando aparece es rojo, cuando desaparece verde, y si es reconocida (que no es el caso), en azul. Luego para las alarmas vamos a utilizar la misma nomenclatura, pero el color de fondo lo pondremos rojo.



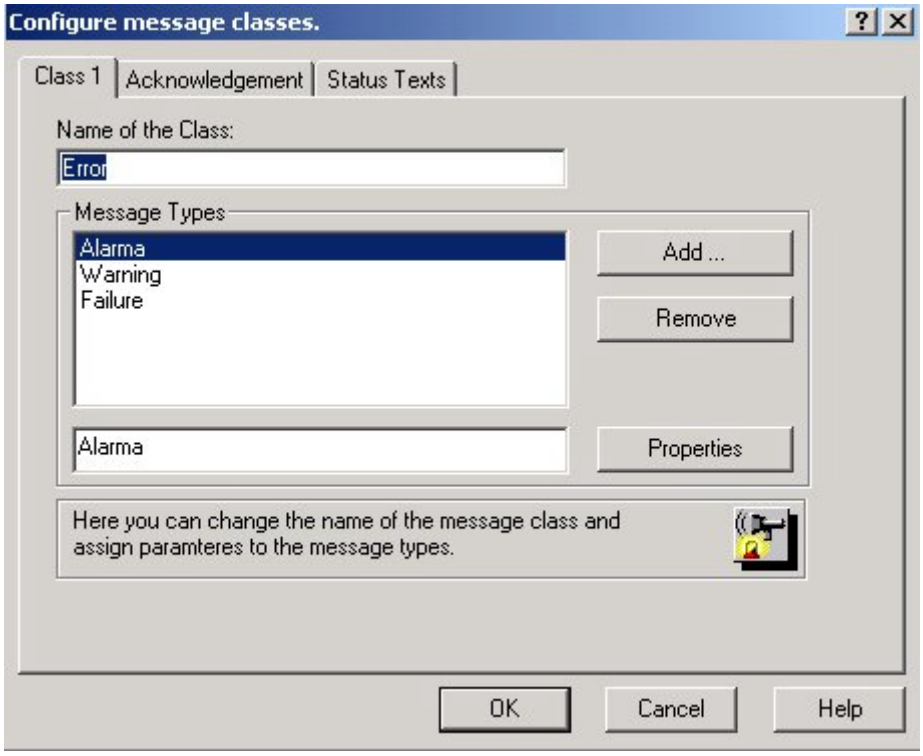
En la solapa reconocimiento, no debemos de marcar acknowledgement CAME in, ya que queremos que sea un aviso, y por lo tanto que cuando desaparezca del plc desaparezca de la pantalla sin mas.



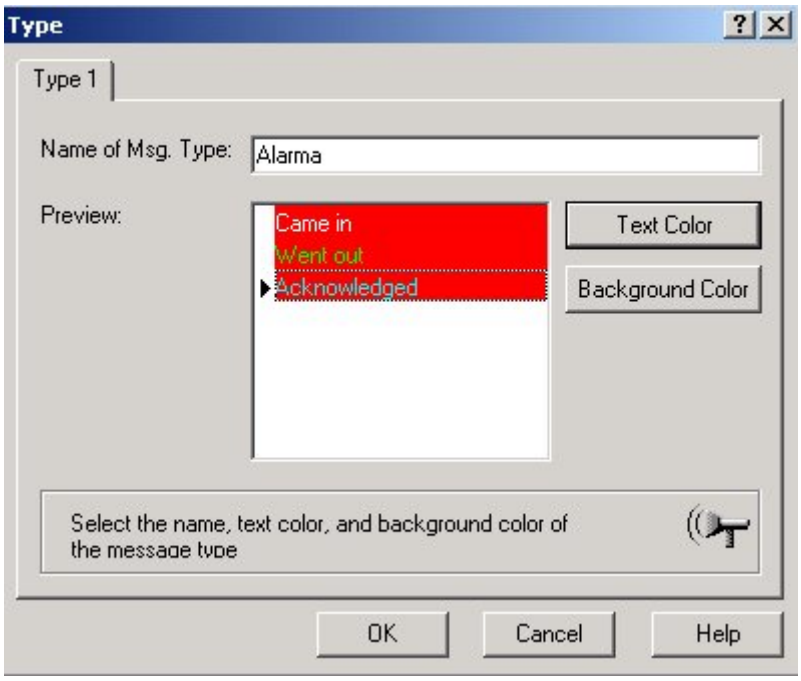
Por último definimos los textos para cuando aparece y desaparece, algo más decente que un mas y un menos que nos pone el sistema.



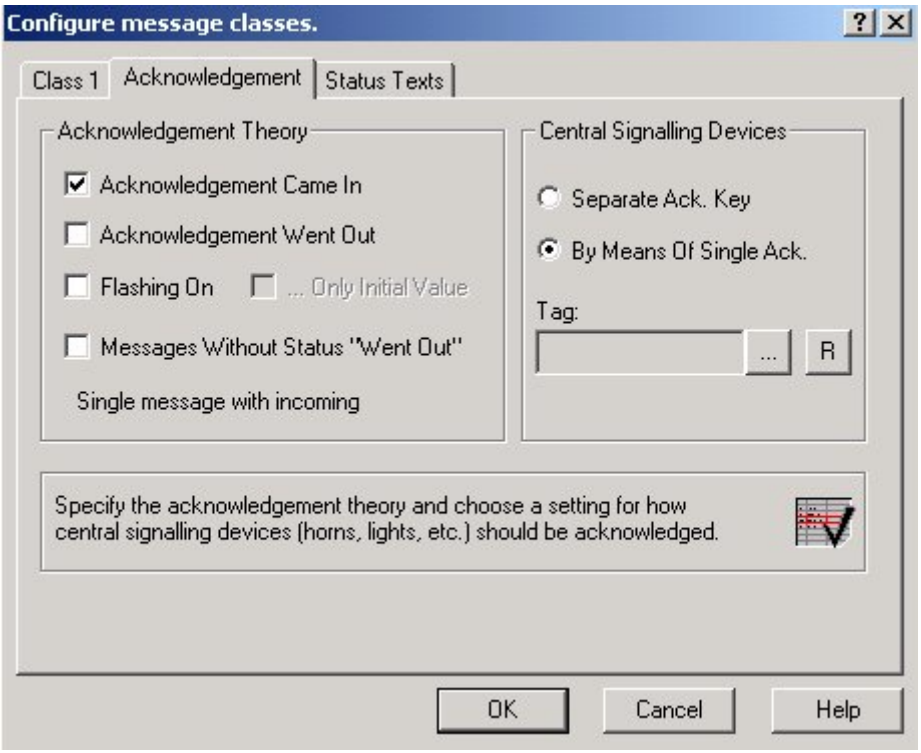
Para las alarmas utilizaremos la clase error, que existe por defecto. Dentro de ella tenemos el tipo Alarm, que cambiamos por alarma, como se aprecia en la figura.



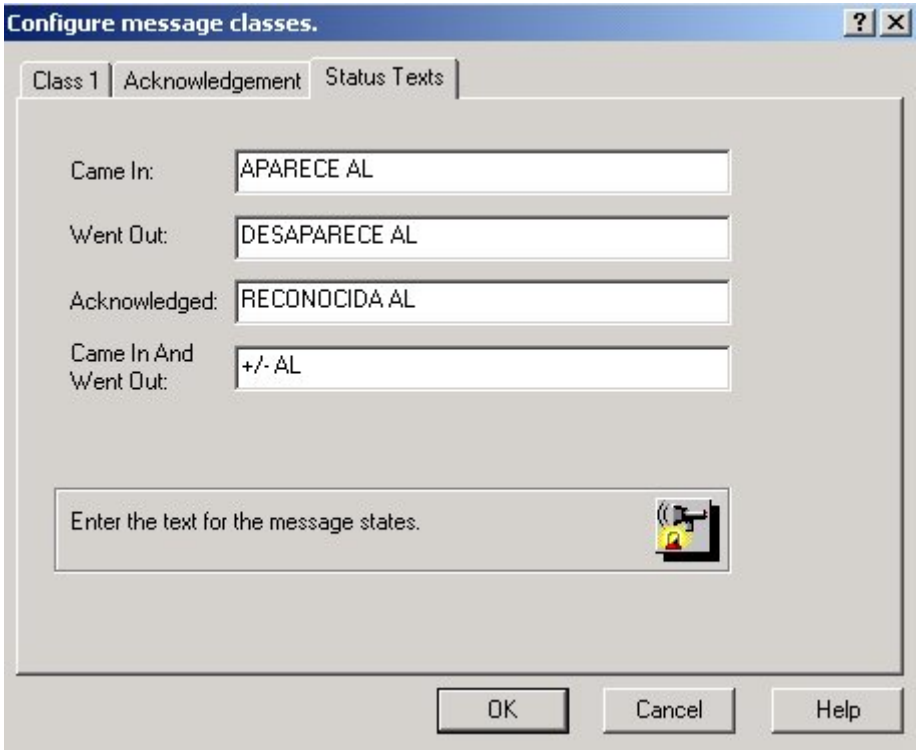
En sus propiedades, cambiamos los colores, para que siempre que aparezca una alarma la línea sea claramente identificativa en rojo, y distinguible del os avisos, que son amarillos.



En reconocimiento podemos observar que las alarmas deberan de ser acusadas cuando aparecen.



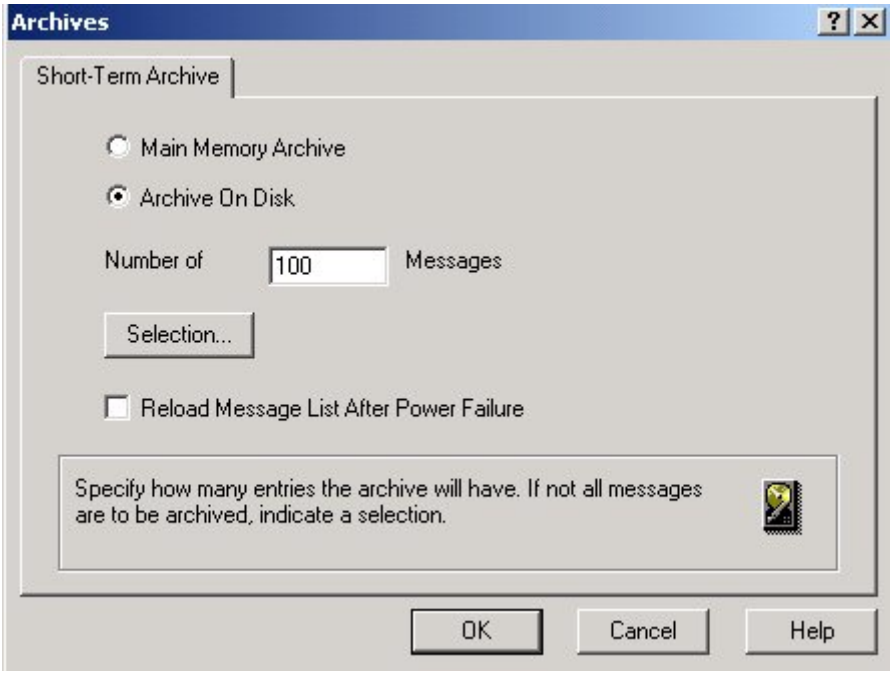
Y los textos de status de las alarmas los sustituimos por los de la figura inferior, mucho mas comprensibles, y diferenciadores de los de los avisos anteriormente creados.



El tema siguiente podría ser group messages, que nos permite agrupar una serie de alarmas o avisos en un grupo, de tal manera que cuando se activen puedan ser acusados todos con un único bit. De momento no vamos a realizar ninguna acción con esto, por lo que pasamos de largo, mas bien de puntillas.

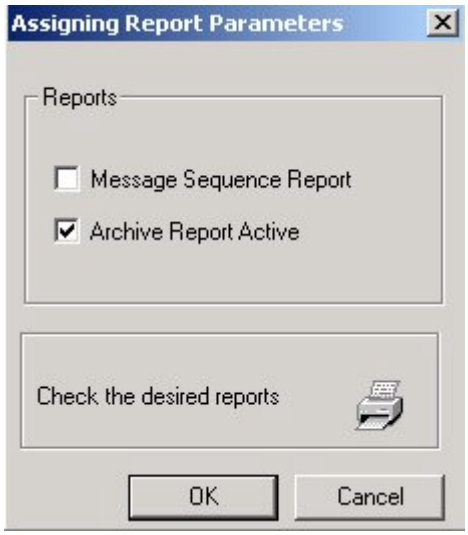
Pero llegamos a archives, que si que es importante. Existen dos maneras de guardar las alarmas en WinCC: en formato tambor (short-term) o a tutti plein[2] (long-term). Siempre, siempre hay que guardar en formato tambor, porque el tutti plein en WinCC le sienta como una patada...

En las propiedades del tambor, definiremos la cantidad de alarmas que debe de almacenar a la vez en el disco duro. En nuestro ejemplo vamos a poner 100, pero eso a gusto del consumidor. A la vez en la pantalla únicamente se podrán ver las última s 1000, aunque en el disco duro el tambor se puede hacer5 mayor. Solución para ver las restantes: filtrar alarmas por fecha o o por tipo (si te suena a chino esto último, tranquilo, que mas adelante se explicará paso a paso).

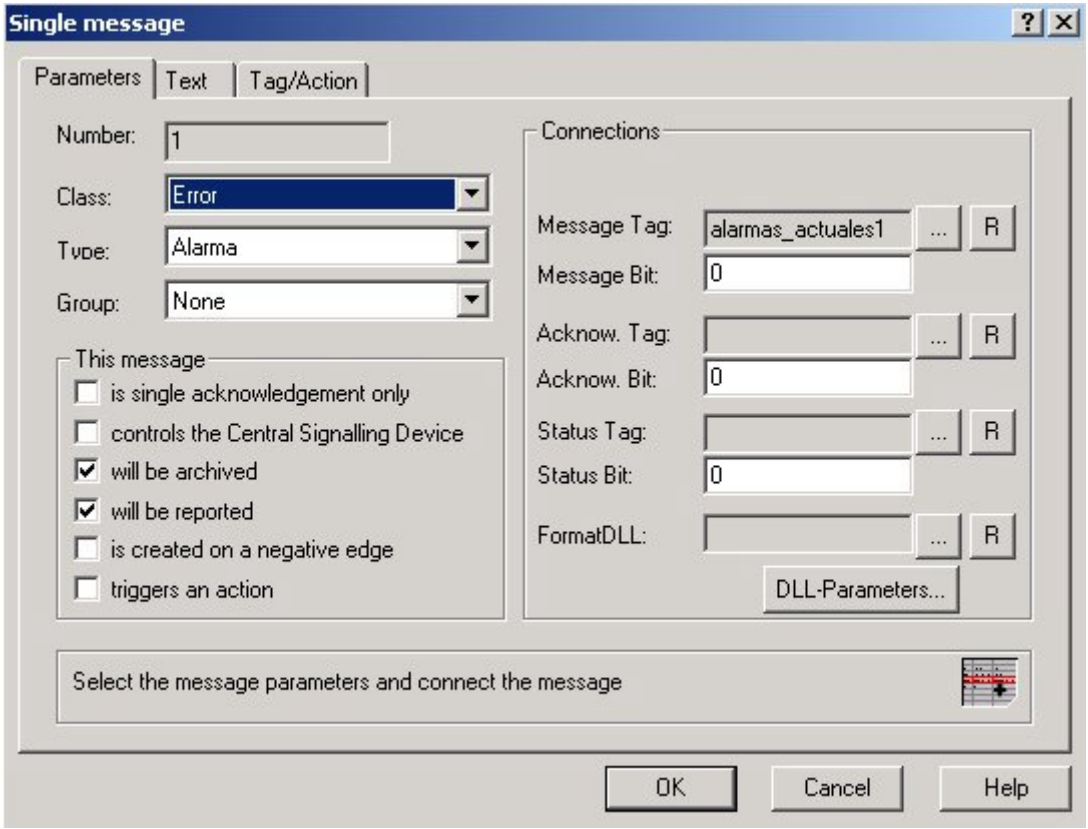


[2] Tutti plein: hasta que reviente el disco duro y los bits se le caigan por los lados.

Ya se acerca el final, que dijo el de Titanic. Nos queda decirle cuando debe de imprimir las alarmas. Podemos decirle que imprima cada vez que salga un mensaje, o solo cuando le demos al botón de imprimir formulario de alarmas. Esta última opción la seleccionamos como se muestra abajo.



Ahora, a generar las alarmas. La primera ya nos la da el sistema. Vamos a modificarla.

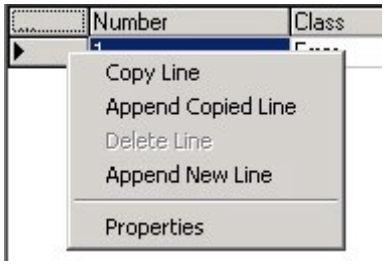


Creamos una variable llamada alarmas_actuales1 de tipo unsigned 32 bits. Con esto disponemos de 32 alarmas en una sola variable de WinCC. Sobra decir que al pagar por variables de comunicaciones, lo que interesa es agruparlas en el máximo tamaño posible, que es dicho tipo. Como se aprecia es de tipo alarma de la clase error, y se activa con el bit 0 de la variable interna. Esto es importante, aunque mas de uno se sentirá ofendido por esta explicación. Si la variable dentro del mapa de memoria del plc es, por ejemplo, la db1.dbd0, el bit que activará este mensaje será el db1.dbx3.0, no el db1.dbx0.0. Si esto no lo tienes claro, no sigas y piensa en como está estructurada la memoria del plc.

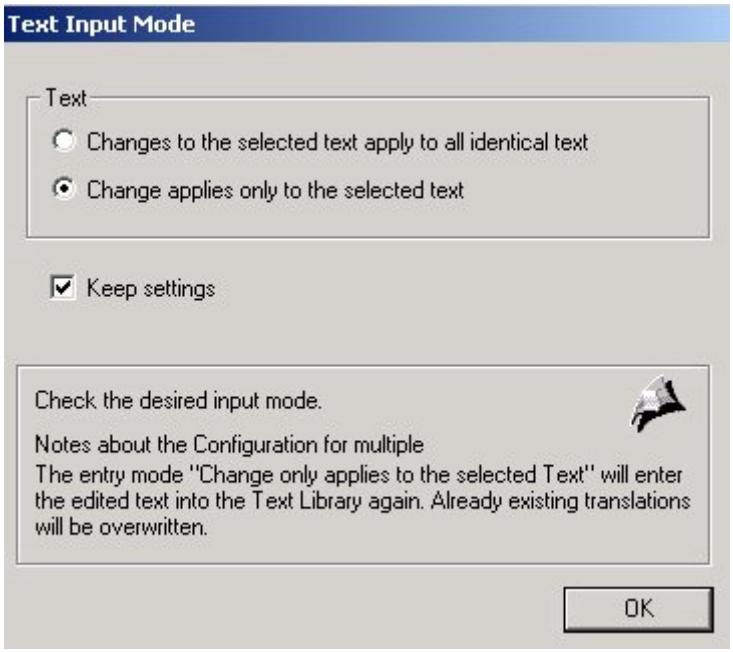
También definimos los textos para los bloques que hemos creado antes. En nuestro ejemplo vamos a poner textos descriptivos, que tu deberías de sustituir por algo así como:

- Infotext: no hay escapatoria, relájate.
- Mensaje: va a estallar el compresor 1.
- Lugar: debajo justo de tus pies.

Bien, hacemos otra alarma, por le método de copiar y pegar la línea anterior, como muestra la figura del menú contextual que aparece al pulsar el botón derecha encima de la línea de alarmas existente.

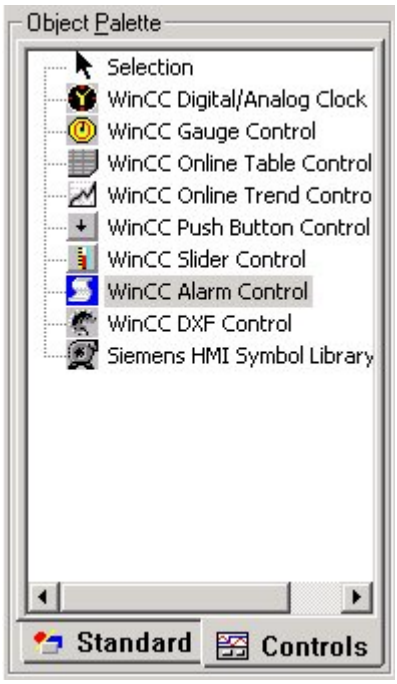


Cuando estemos hacie3ndo una segunda alarma, que asignaremos a la misma variables alarmas_actuales1 de antes, pero esta vez al bit 1, nos aparecerá un mensaje muy simpático como la figura inferior: Debemos de seleccionar cambiar solo en el texto seleccionado, porque sino realizará las modificaciones no solo en la línea de alarmas que estemos editando, sino en todas las del mismo tipo (si no te aparece el mensaje, no te preocupes, que todo lo malo llega inexorablemente). Para que no te vuelva a preguntar esto mismo cada vez que modifiques una alarma, recuerda seleccionar la casilla sep settings.



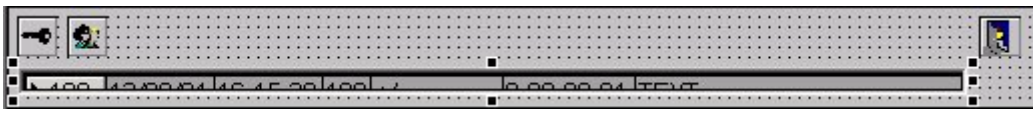
Como ya tenemos dos alarmas, para que mas, nos salimos guardando del alarm logging. Ahora tenemos que definir la ventana dentro de una picture donde deseamos observar las alarmas. Para ello entramos al graphic designer, y, ¿en qué picture la pondremos?. Lógicamente en una que esté en todas las pantallas, así podremos saber si una alarma se encuentra activa independientemente del lugar en el que nos encontremos en el scada. ¿Dónde será?. Supongo que habrás adivinado ya que es en bottom.pdl.[3]

Deberemos de seleccionar el control WinCC Alarm de la solapa controls del object palette.

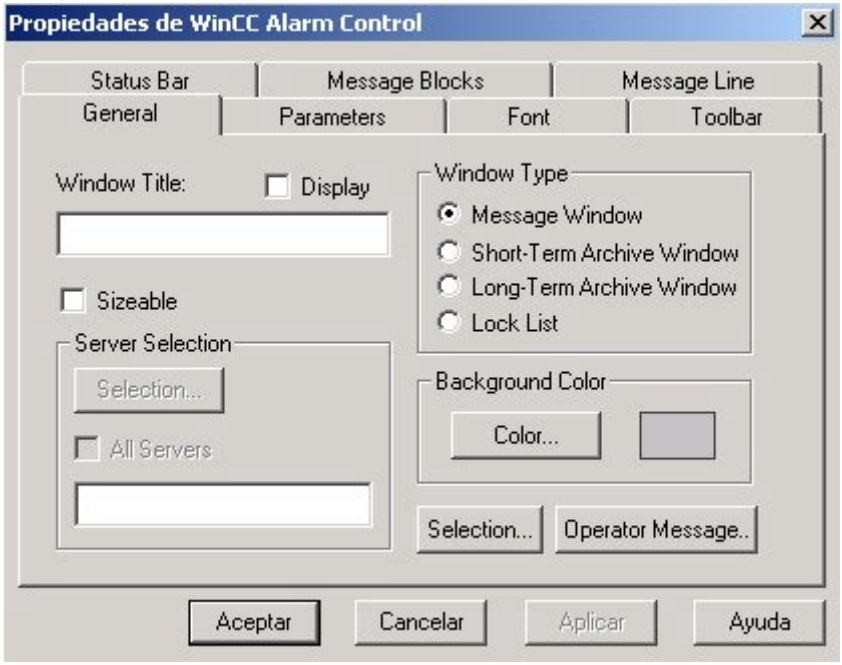


[3] Si no lo habias averiguado solo te diré las sabias palabras del cocinero del Titanic: “The thing is very bad”.

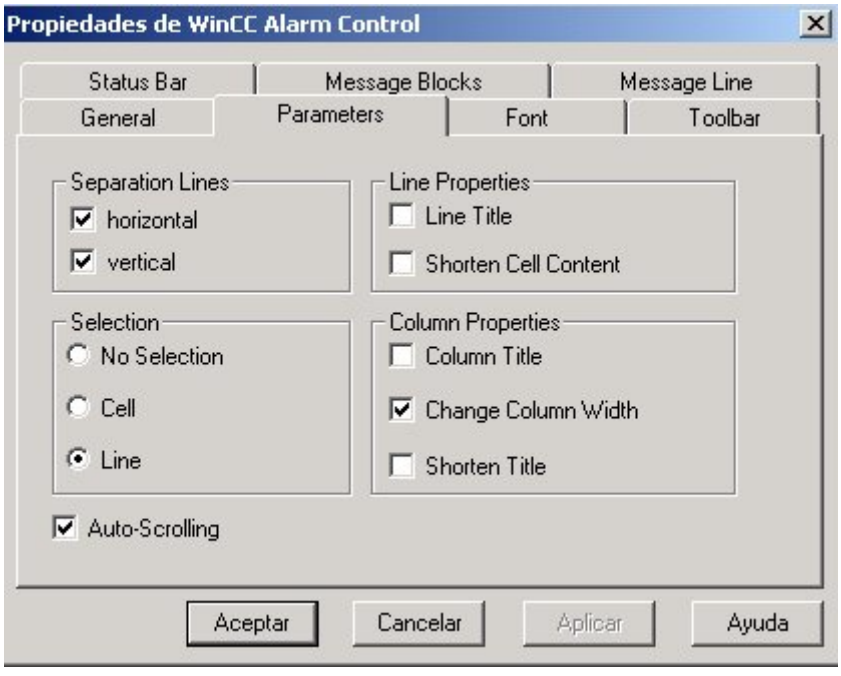
Si no te aparece a la derecha el object palette es que te lo has quitado. Vete a view->toolbars y activala. Ponemos dicho control en la pantalla del bottom.pdl como muestra la figura:



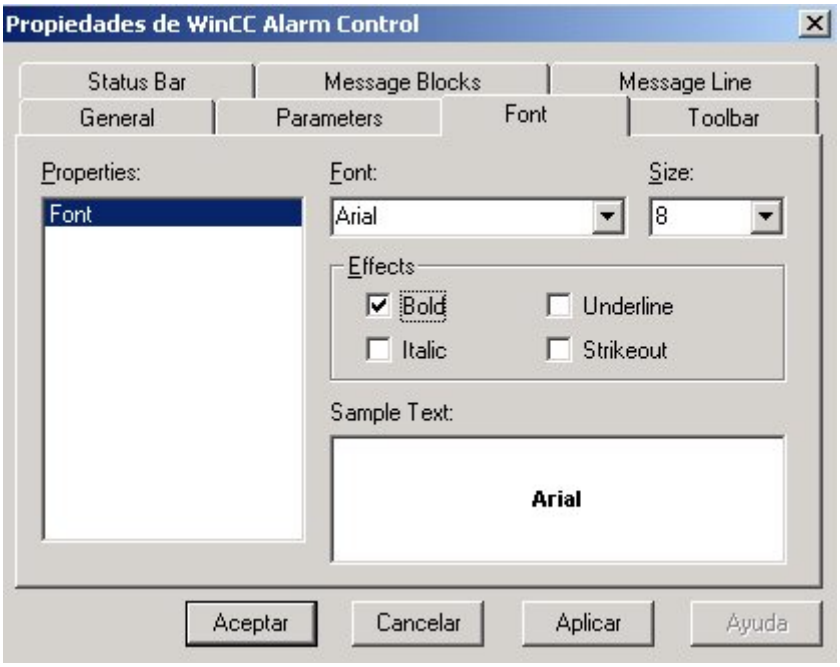
Pinchando dos veces sobre el control o ventana de alarmas que acabamos de colocar, nos aparece el siguiente cuadro de diálogo:



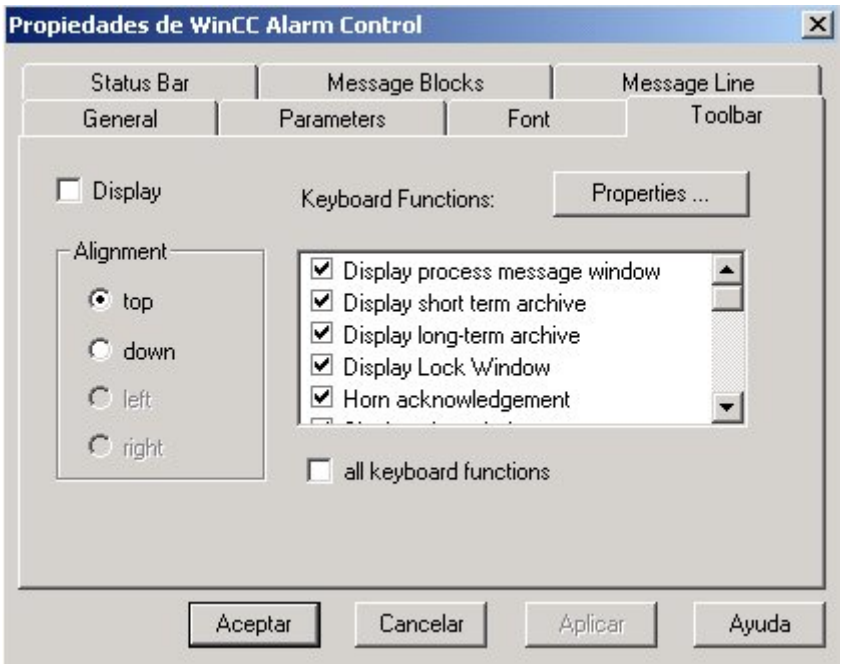
Quitamos el display de window title, nada de sizeable,y lo mas importante: el window type es message window, para que aparezcan las últimas alarmas de la instalación.



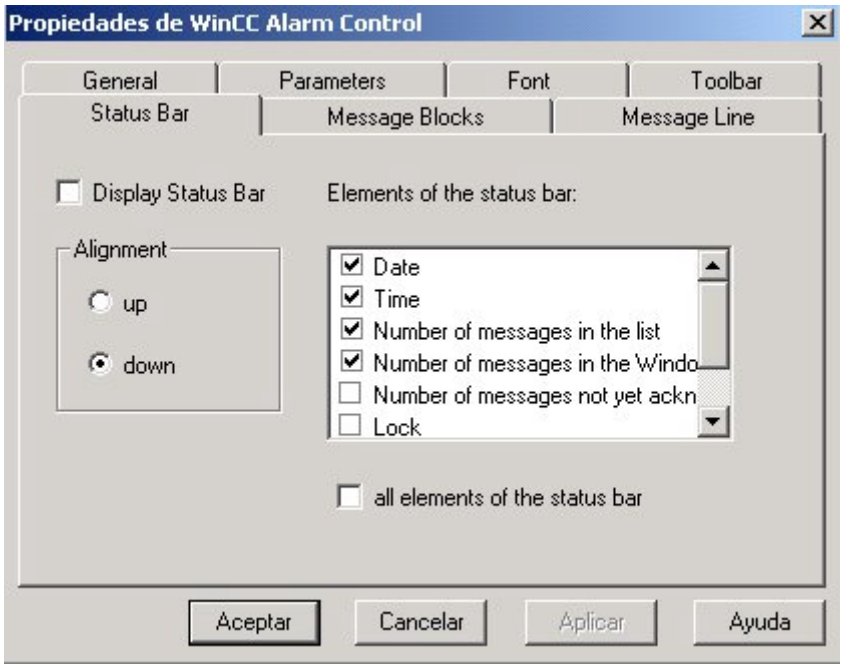
En la siguiente solapa nos viene el tema de propiedades de línea. Le quitamos el titulo a las lineas, y a las columnas. La selection será del tipo line. En auto-scrolling definimos si deseamos ver la última alarma, o la primera que apareció y continua activa. Normalmente la gente quiere saber lo último que está pasando, por lo que la activamos.



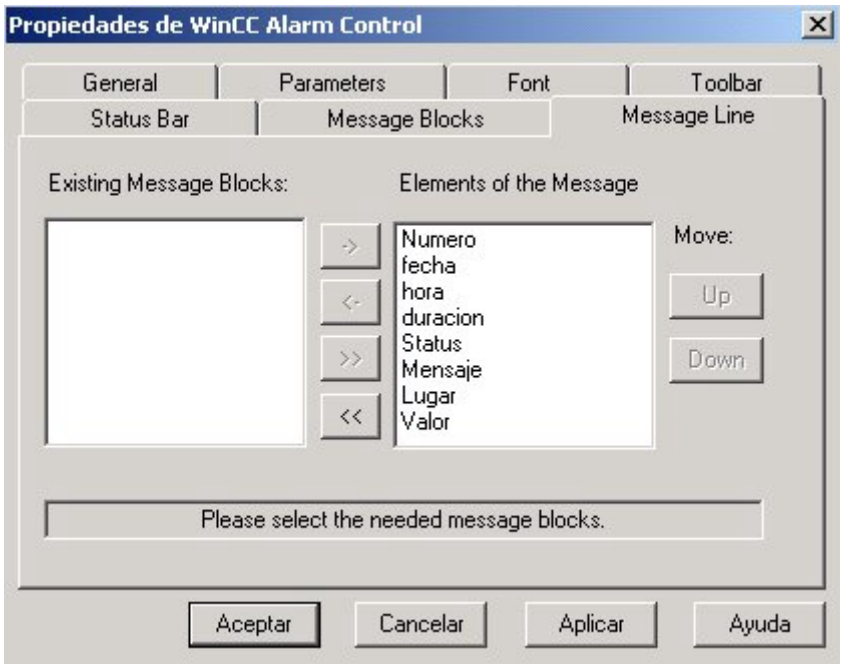
La fuente le ponemos la más pequeña que tengamos y en negrita, para verla bien.



La barra de herramientas o toolbar no la vamos a mostrar. Entonces, te preguntarás: cómo vamos a acusar las alarmas si no tengo barra de herramientas?. Pues con un botón externo que nos colocaremos. C forever. Continuamos.



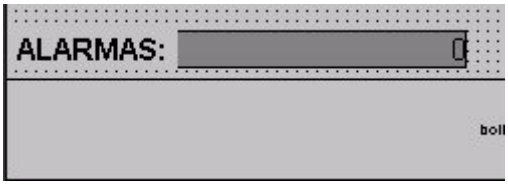
Status bar tres cuartos de lo mismo: nada de nada, ni verlo.



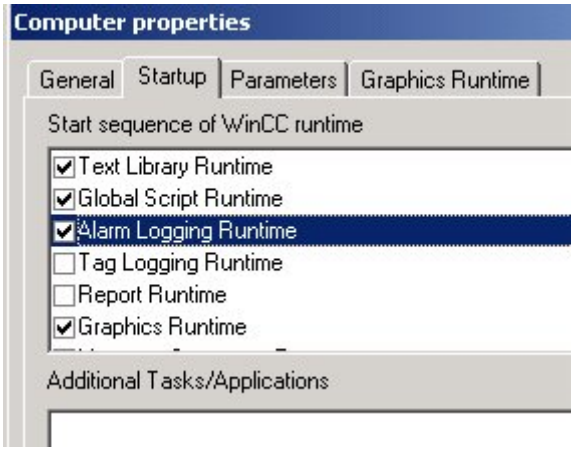
Y por último nos queda la línea, que tendrá los bloques que definimos anteriormente. Los ordenamos según deseemos que aparezcan en la línea de alarmas.

Con esto hemos acabado. Si arrancamos nuestro proyecto, no irá. Que bien. Pues a casa, ¿no?. ¡No!. El tema si has llegado hasta aquí está controlado (aunque tu no lo sepas ni lo creas).

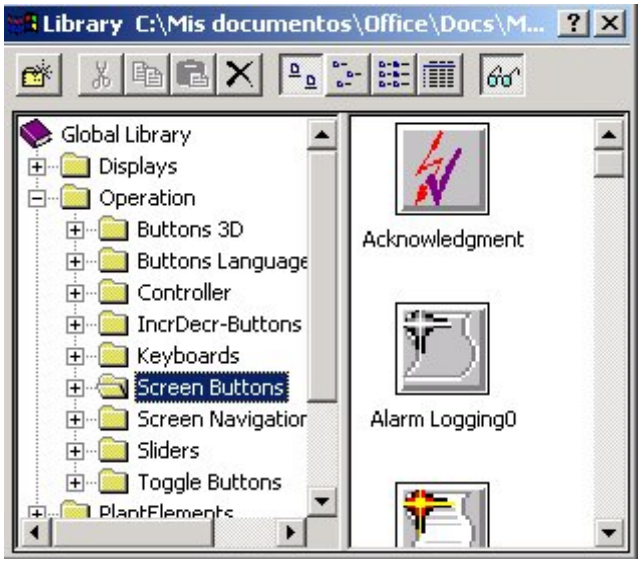
Ahora nos vamos a la ventana start.pdl y colocamos una i/o de tipo both (entrada/salida), en formato binario y asociada a la variable alarmas_actuales1, mas o menos dentro de la picture start.pld, un poco por encima de bottom, como muestra la figura:



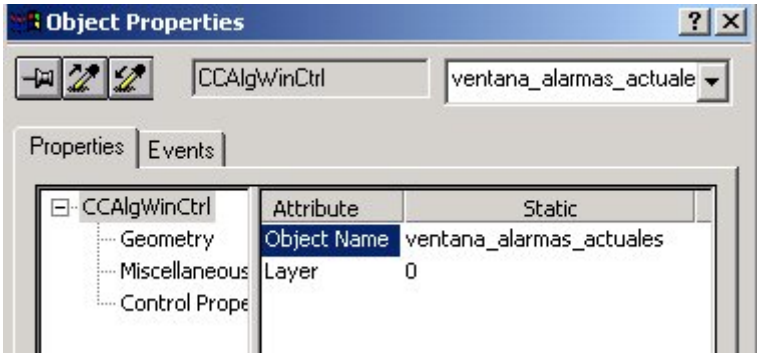
Ahora nos salimos del graphic designer, y activamos la parte de runtime del alarm ligging, dentro de las propiedades de nuestro pc. Si no sabes de qué diablos te estoy hablando, mira el control center, mira Computer, y lo que hay a la derecha, ¡eso es, nano, por fin!.¿Ha costado, eh?. Bien, prosigamos.



Con esto tenemos: una ventana en la que aparecen las alarmas que activamos en la i/o de la pantalla start.pdl. Como tenemos dos definidas, con un 1 activamos la primera y con un 10 (recuerda que es binario) activamos la segunda. Vale, pero, y ahora ¿como acusamos las alarmas?. Porque si vienen dos, no voy a poder ver la segunda en la vida. La solución pasa por poner un botón de acuse a la derecha de la ventana de alarmas que acabamos de realizar. Volvemos a abrir el graphic designer con la picture bottom.pdl. De la librería de winCC seleccionamos el que aparece en la pantalla.



Ahora a ese botón le vamos a asociar un código que permita acusar las alarmas. Pero antes tendremos que realizar un pequeño paso necesario. Selecciona la ventana de alarmas dentro de bottom.pdl y en sus propiedades, cambiale el nombre a algo comprensible, como por ejemplo ventana_alarmas_actuales (ver figura).



Ahora vuelve a seleccionar el botón que hemos cogido de la librería antes, y en events, en el evento mouse actino, vamos a añadir el siguiente código:

AXC_OnBtnSinglAckn("bottom.PDL","ventana_alarmas_actuales");

Con esto ya podremos acusar desde cualquier ventana las alarmas. Nota: si se te ocurre modificar las propiedades de la ventana de alarmas a posteriori , el wincc tiene un bug de programación que hace que no acepte nuevas modificaciones de algunos de los parámetros de la ventana de alarmas una vez configurada. Lo mejor que puedes hacer es borrarla y volverla a crear (si, ya se, menuda mi...fin del capítulo).