Capítulo 5.

Metodología propuesta para la integración de los sistemas de comunicaciones de campo.

5.1 Descripción del tipo de problemas que se pretenden solucionar.

La integración de elementos de campo heterogéneos, es decir con protocolos o sistemas de comunicación diferentes, tales como autómatas, dispositivos de E/S, etc., normalmente significa interpretar la forma de transferir los datos que cada uno de ellos implementa, y enviar estos datos al sistema integrador.

La complejidad de la integración dependerá principalmente de la cantidad y distribución espacial de estos elementos. Este trabajo está orientado a servir de alternativa de integración en los siguientes casos:

1. En todos aquellos sistemas que ya cuentan con elementos heterogéneos distribuidos a lo largo de toda la planta y de los que se requiere su integración para su control o supervisión. Un ejemplo son los sistemas que han crecido por aumento de producción (caso de líneas de fabricación construidas en fases sucesivas).

2. En los sistemas que están en la fase de diseño pero que, debido a las imposiciones de los fabricantes de equipamiento industrial, resulta imposible adoptar un único estándar de comunicación. Esto ocurre comúnmente porque en los equipamientos industriales normalmente se implementa un sistema de comunicación que se adecua a los requerimientos del fabricante y no a los del usuario.

3. En aplicaciones que dispongan de *drivers* de comunicación con los dispositivos, pero donde las distancias físicas de separación, de éstos al elemento de integración, impiden el uso de las interfaces estándares de los mismos. Normalmente el elemento integrador será un PC, cuyas interfaces estándares - generalmente RS232 -, no son capaces de alcanzar las distancias usuales dentro de los entornos automatizados. En la Figura 41 se ilustra esta situación.

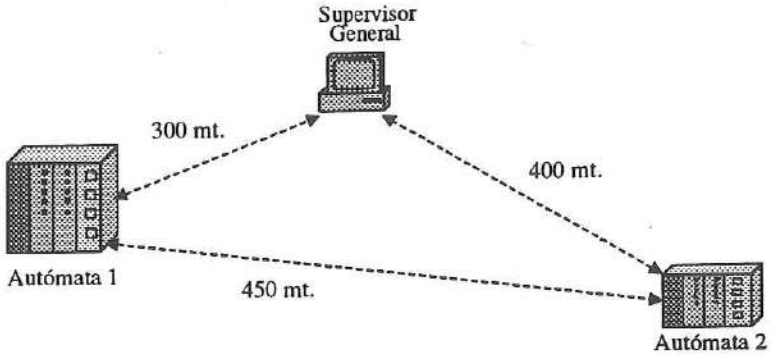


Figura 41: Ejemplo de sistema a integrar.

4. En las aplicaciones donde algunas variables requieren ser transferidas cumpliendo restricciones, por ejemplo de tiempo o de consistencia, que los sistemas de integración usuales no garantizan.

5. Casos en que los elementos a integrar soportan una interfaz propia de comunicación, por ejemplo autómatas que tienen capacidad de utilizar interfaces para buses de campo - como por ejemplo FIP, u otros protocolos -, pero donde la cantidad y flujo de datos que se desea exportar no justifican el uso de los mismos; esta justificación puede ser tanto en costo por nodo como por eficiencia del bus. En la Figura 42 se muestra de manera gráfica esta situación, para dos autómatas que soportan un bus propio.

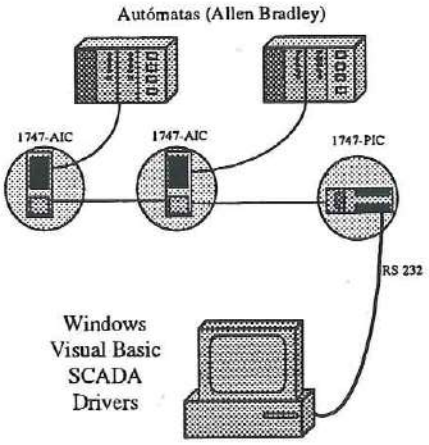


Figura 42: Integración de elementos con interfaces de comunicación propias.

6. Los casos en que la fabricación de los elementos de comunicación ha sido discontinuada por los fabricantes, o el mantenimiento resulta costoso por el tipo de hardware utilizado. En muchas aplicaciones de hace pocos años, las interfaces de comunicación de los autómatas y otros dispositivos, se realizaban según protocolos propietarios o siguiendo una tecnología particular que, hoy en día, resultan muy costosas, tendiéndose normalmente a discontinuar su producción.

7. Integrar equipos o procesos aislados, cuyas especificaciones iniciales no contemplaban el trabajar conjuntamente, con otros elementos dentro del entorno.

8. Los casos que son una mezcla de las anteriores, cuya integración siempre será posible con la metodología propuesta en esta tesis.

De forma más general puede afirmarse que la probabilidad de que todos los elementos a integrar dentro de una planta ya en producción, tengan capacidad de implementar el mismo sistema de comunicación es muy pequeña. Por consiguiente es un reto industrial frecuente el tener que **intercomunicar elementos que no se "entienden"**.

5.2 Planteamientos iniciales.

Cuando se plantea la integración de los sistemas de comunicación de campo de una planta industrial, generalmente encontramos que ésta se encuentra automatizada por líneas de producción o por zonas específicas; encontrándonos con "islas" de dispositivos, agrupados de acuerdo a las necesidades específicas de la parte del sistema que automatizan. Esto se debe a que las necesidades de integración normalmente se presentan cuando la planta ya está en funcionamiento desde hace algún tiempo, o bien por necesidades de mejoras de la productividad, o más aún, porque las nuevas tecnologías obligan a aumentar las prestaciones de los sistemas de comunicación, en todos los niveles, y para la totalidad del sistema productivo.

Cuando se diseña una nueva planta industrial se pueden planificar los sistemas de comunicación en todos los niveles de la pirámide CIM, estudiada en el apartado 4.2.1, pero esto es una tarea muy difícil de conseguir a nivel de campo. Lo que sucede en este nivel es muy diferente de lo que ocurre en los niveles superiores, donde los sistemas de comunicaciones se encuentran perfectamente definidos, pues muchos fabricantes de maquinaria y sistemas industriales utilizan elementos de campo cuyos sistemas de comunicaciones no serán compatibles con las especificaciones del diseño. Dicho de otra manera, encontramos muchos elementos, necesarios dentro del sistema productivo, que no se adecuan a los sistemas de comunicaciones que de forma "*top down*" se especifican.

Muchos de los fabricantes de dispositivos de campo, pero en absoluto todos, expresan sus buenas disposiciones para fabricar elementos de campo completamente estandarizados. El problema es que el comité técnico número 65C, designado por el IEC *(International Electrotechnical Committee)* para estandarizar los buses de campo, encuentra muchas dificultades para lograr un consenso al respecto. La realidad del sector muestra claramente que va a ser muy difícil lograr tal consenso. Las propuestas iniciales para la estandarización - el estándar Alemán PROFIBUS [DIN19245] y el Francés WorldFIP [WorldFIP] - tienen varios puntos en los cuales no hay acuerdo. Estas propuestas se vieron incrementadas, en 1995, por la propuesta del estándar Danés

P-NET [P-NET] quedando mucho camino por recor para lograr un acuerdo definitivo. Es más, una vez definido tal estándar universal, si es que una vez llega a existir, pasarán muchos años antes de que se pueda decir que la mayoría de los sistemas industriales automatizados son interoperables y 100% intercambiables en el nivel de buses de campo. Ciertamente el estándar es deseable, y beneficiará a los usuarios finales, que podrán elegir los dispositivos que más se adecuen a sus requerimientos, sin estar limitados a que cumplan uno u otro protocolo. Pero ningún estándar podrá mantenerse estático mucho tiempo, siendo previsible su evolución y su sustitución por otros.

La evolución ocurrida en las redes informáticas y de telecomunicación es fuente de inspiración para buscar soluciones en el campo industrial. Por un lado están los planteamientos reguladores y estandarizadores ("*top-down*") que promueven la homogenización acordada de los sistemas. Esto se ha demostrado eficaz hasta cierto límite. Por otro lado, de forma complementaria, se han desarrollado técnicas sofisticadas de interfaz (*bridges, routers, gateways*, etc.) entre redes muy diversas (LAN Ethernet, LAN Token Ring, SNA, DECNET, RDSI, SDH, ATM, etc.). El concepto clave en este segundo enfoque es la concentración y la gestión de las interfaces y de los protocolos para la interoperación entre redes. La concentración incluye con frecuencia la introducción de una "red de interfaz" intermedia. La evolución de las redes informáticas ha demostrado que estos elementos o equipos de interfaz también evolucionan con las redes, pero su necesidad no desaparece.

5.3 Metodología propuesta.

En la presente tesis proponemos una metodología sistemática y flexible, orientada a la integración de elementos heterogéneos de campo, autómatas, buses de sensores, etc., que requieran ser integrados dentro de los niveles superiores de las comunicaciones.

Esta metodología – al igual que el enfoque "*top down*" integrador por vía de estandarización- requiere una completa planificación "a priori" de los sistemas, pero en cambio utiliza un enfoque "*bottom up*" integrador por vía de interfaz de mayor flexibilidad y que resuelve también la integración "a posteriori" de sistemas.

Para lograr la integración según la metodología propuesta especificaremos un dispositivo inteligente, Concentrador de Comunicaciones de Campo (CCC), que se configure fácilmente, que se conecte por una parte a los dispositivos de campo, y por otra permita conectarse a los sistemas de comunicación de mayor nivel. La conexión con los sistemas de mayor nivel, cuando existe una red de CCCs, será realizada por uno de ellos que se denominará CCC principal o CCCP que integra todos los datos del sistema. La Figura 43 muestra gráficamente tres CCCs formando una red de integración.

Las redes formadas con este dispositivo deberán también permitir exportar variables a otros elementos del sistema, pero garantizando la consistencia temporal de las mismas, y respetando además las restricciones de tiempo existentes.

Para la mayoría de los casos de integración de elementos de campo heterogéneos, esta metodología constituye como veremos una solución eficaz, a la problemática de la integración presentada en los capítulos 2, 3 y 4.

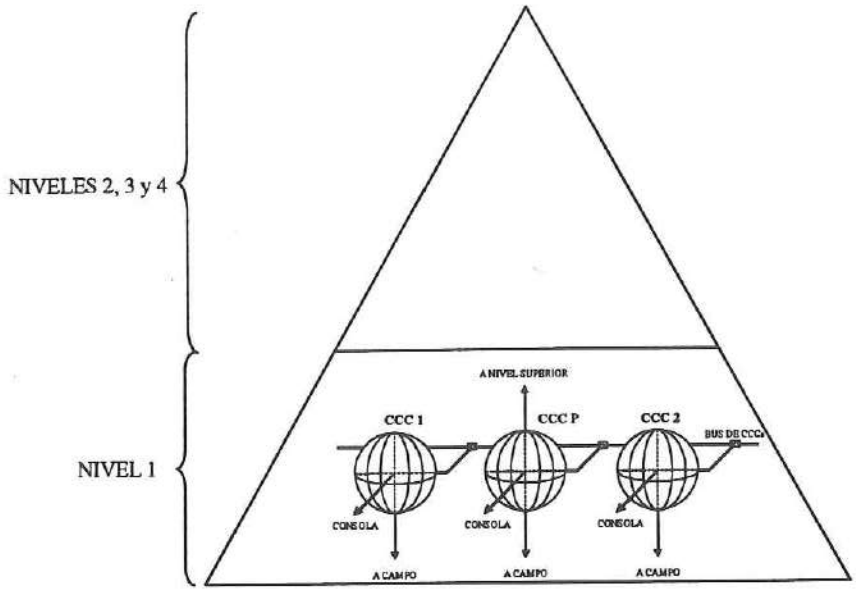


Figura 43: Red de integración con CCCs.

La información en los Concentradores de Comunicaciones de Campo (CCC) se presentará de forma transparente para todos los usuarios del sistema que se encuentren conectados a él.

Los objetivos que se pretenden lograr con la presente metodología son los siguientes:

1. Permitir la comunicación bidireccional y transparente entre todos los dispositivos de campo que así lo necesiten, sin importar la red ni el protocolo que soporten.

2. Garantizar los tiempos, y las restricciones de consistencia temporal, en los mensajes críticos entre nodos CCC, cuando se necesiten transferencias en tiempos máximos definidos.

3. Permitir el intercambio de datos bidireccionales con los otros niveles del sistema de comunicación.

4. Permitir la configuración remota al inicializar el sistema. Cuando existan más de un CCC en una red permitir la configuración remota de los nodos, desde el que será el CCC principal, o bien desde los niveles superiores.

5. Definir una arquitectura para el CCC compacta, modular y de bajo coste para integrar y facilitar las comunicaciones en entornos industriales.

5.4 Tipos de elementos y sistemas a integrar.

Seguidamente analizamos los tipos de elementos y sistemas a integrar en sus configuraciones básicas, señalando no obstante que se pueden encontrar algunos que son híbridos.

1. Elementos de Entrada / Salida discretos.

   Este es el caso de muchos de los sensores y actuadores discretos (*ON/OFF*), entre los cuales podemos citar fines de carrera, presostatos, cilindros neumáticos, etc. Estos elementos se conectarán directamente a los puertos de E/S discretas del CCC y en la configuración del mismo se les asignará un número de identificación, comprendido entre cero y 65535, y un nombre de hasta 16 caracteres.

2. Transductores y actuadores no discretos.

   Generalmente los transductores y dispositivos de campo que necesitan interconexión analógica (sensores de presión, humedad, nivel, etc.) o especial (por ejemplo *encoders* ópticos), son conectados directamente a los autómatas o lazos de control local. Aunque también muchos de ellos pueden ser conectados directamente a los puertos disponibles en el CCC, que pueden configurarse pin a pin para tal fin.

3. Autómatas individuales.

   Este es el más común de los casos, ya que la mayoría de los autómatas, y así seguirá por muchos años, implementan para intercambio de datos un puerto serie normal. Este puerto sirve a su vez de puerto de programación. Gracias a la tendencia de los sistemas abiertos es posible adquirir los *drivers* oportunos a bajo costo. Hay que tener en cuenta, cuando se definan variables con restricciones de tiempo, que los autómatas que las tengan asignadas permitan transferencias, tanto de lectura como de escritura, compatibles con las restricciones de tiempo que se pretende asignar. Es decir las restricciones de tiempo no serán cumplidas si los elementos finales, en este caso los autómatas, no lo permiten.

4. Autómatas formando redes propietarias.

   Este es el caso de casi todos los autómatas de bajo y algunos de medio nivel, pues los fabricantes han desarrollado, y continuarán desarrollando, protocolos propios, de acuerdo a las características y tamaños de las transferencias que realizan. De esta manera optimizan su hardware en tamaño y coste. Es necesario aclarar que la implantación de protocolos de alto nivel, como PROFIBUS, FIP, etc., requieren una capacidad de procesamiento notable, por lo que éstos son utilizados en elementos de prestaciones altas. También hay que recordar que en el bajo nivel los intercambios tienen tamaños reducidos. Por ejemplo la red de sensores ASI (*Actuator Sensor Interface*) utiliza mensajes de sólo 4 bits.

5. Sistemas de control propietarios.

   Muchas aplicaciones de control se realizan empleando algunos sistemas de comunicación particularmente aptos a los requerimientos específicos, como por ejemplo las redes de riego. En este tipo de sistemas el protocolo es simple, normalmente maestro / esclavo, y se realiza con interfaces especializadas, por ejemplo formando una red como *Meter Bus de TEXAS INSTRUMENT*, u otro tipo particular para estas aplicaciones. Estos buses incluyen normalmente, sobre los dos cables de conexión, la alimentación de los dispositivos remotos.

Si bien este tipo de sistemas no tiene restricciones de tiempo, no pueden ser dejados de lado al estudiar la integración de sistemas de comunicación, pues son muy comunes sobre todo formando parte de los macrosistemas de gestión total de agua.

6. Controladores locales.

   Cuando existan aplicaciones de control local, por ejemplo PIDs, controles todo /nada, etc. estos también podrán ser integrados mediante el puerto de comunicación que tengan disponible.

5.5 Concentrador de comunicaciones de campo CCC.

El CCC será un dispositivo inteligente conformado por una CPU y múltiples puertos de comunicación, para conexión con los elementos de planta y con los sistemas de comunicación superiores. Además este dispositivo deberá implementar:

1. Los protocolos de los elementos de planta, que se encuentran conectados a él por medio de puertos dedicados.

2. Un protocolo de intercambio de información entre CCCs, que se encuentren interconectados al mismo nivel.

3. Los mecanismos internos para coordinar el intercambio de datos.

4. Un protocolo para conexión con el nivel superior.

Este dispositivo es el elemento principal para lograr la integración de las comunicaciones de campo con la metodología propuesta. En la Figura 44 se ilustra gráficamente el nodo CCCP (este nodo sólo se diferenciará de los otros por la conexión que realiza con los sistemas de comunicación superiores). En esta figura se puede apreciar que las variables del sistema serán recopiladas en todos los CCCs como objetos, para su transmisión eficiente en la red de CCCs. El CCCP deberá transferir a los niveles superiores las variables de todo el sistema integrado.

5.5.1 Requerimientos mínimos del CCC.

Presentamos a continuación las principales características, tanto hardware como software, que debe presentar un CCC.

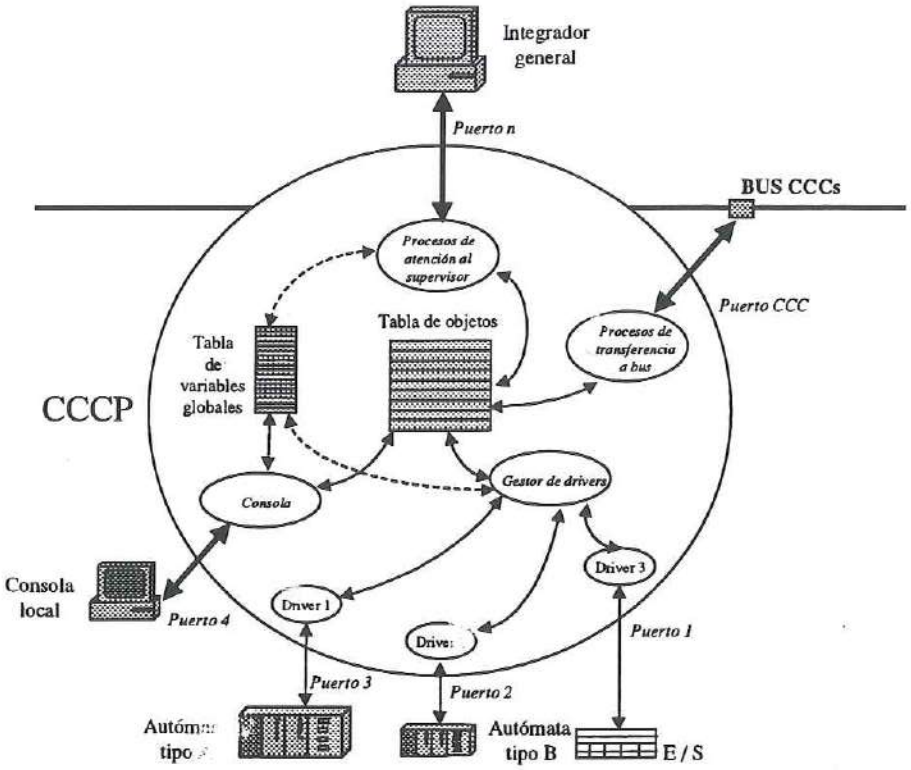


Figura 44: Nodo CCC del sistema propuesto.

5.5.1.1 Requerimientos hardware.

Los requerimientos hardware más importantes del CCC, son los siguientes:

1. Máquina basada en microprocesador con capacidad de incorporar sistema operativo en tiempo real.

2. El sistema operativo y los programas deberán almacenarce en memoria no volátil como por ejemplo EPROM o FLASH. El CCC debe contar también con memoria tipo SRAM o DRAM para gestión del sistema operativo y de los procesos. Las cantidades mínimas se indican en la Tabla 17.

Tipo de CCC	EPROM / FLASH	SRAM
CCC	128 k	128 k
CCCP	512 k	128 k

Tabla 17: Requerimientos de memoria mínimos recomendados para un CCC.

Se ha descartado el uso de disco duro a este nivel, ya que el entorno de funcionamiento de los CCC será el nivel de planta donde existen vibraciones y otros factores que desaconsejan su uso, así como por la pérdida de la versatilidad de los CCC si su coste aumenta.

3. Posibilidad de incorporar dispositivo de almacenamiento no volátil, por ejemplo PCMCIA, para guardar configuración o históricos de alarmas / eventos.

4. Puertos de comunicación: La cantidad y tipo de elementos – autómatas, E/S, etc. - que debe integrar, determinará el número y tipos de puertos de comunicación que debe tener para este fin. Otros puertos de comunicación presentes en el CCC se indican en la Tabla 18.

Tipo de puerto	CCC	CCCP
Consola (RS-232)	si	si
Comunicación entre CCCs	si	si
Ethernet	no	opcional
Módem (RS-232)	no	opcional
Bus de campo	no	opcional

Tabla 18: Puertos de comunicación en el CCC.

5. Puerto de entradas / salidas: Como en muchas aplicaciones se necesitará integrar dispositivos de entrada / salida aislados (digitales o analógicos) el CCC debe tener un puerto analógico y un puerto digital de 16 bits configurable bit a bit.

5.5.1.2 Requerimientos software.

Respecto al software los requerimientos más importantes son:

1. El sistema operativo de los CCCs será un sistema operativo multitarea en tiempo real. Ésto se hace con el fin de garantizar la ejecución de las tareas específicas del CCC, atención de *drivers*, mantenimiento y actualización de los objetos de comunicación, etc., y para garantizar los tiempos máximos de las tareas que lo necesiten. Por otro lado el sistema operativo debe ser de pequeña dimensión, "ligero", de bajo costo y altas prestaciones, de forma que el CCC no pase de ser un elemento auxiliar de la red.

2. El sistema operativo debe permitir su instalación en EPROM. A ser posible deberá ser de amplia difusión comercial para facilitar la adquisición de drivers.

3. El software debe permitir la carga dinámica de nuevos módulos dentro del sistema.

4. El software de configuración de las variables del sistema y de los puertos de comunicación debe ser simple y flexible.

5. En una red de CCCs el nodo principal debe ser capaz de autoconfigurar todo el sistema. Además se debe permitir a los nodos abandonar o darse de alta en la red sin interrumpir el servicio.

6. Se deben incluir todos los drivers de comunicación que se necesiten para el sistema dado. La activación de los drivers se realizará por software de configuración.

5.5.2 Funciones del CCC.

Las principales funciones que desempeña el CCC son las siguientes:

1. Único nodo de interconexión con los niveles superiores o con el SCADA (*Supervision Control And Data Acquisition*).

   Como en el CCCP se tiene toda la información de los elementos de campo que integra, éste puede ser conectado al sistema SCADA, o al nivel superior, como nodo único. El SCADA lo reconocerá como un sistema completo en donde encuentra todos los datos, con un formato determinado, necesario para su funcionamiento. Esta forma de conexión aumentará las prestaciones del sistema de supervisión y control al liberarlo de las operaciones de gestión de los protocolos de comunicación con los *drivers* de los dispositivos de campo que tenía que realizar bajo un esquema convencional de integración. Igualmente las ordenes de control las enviará el SCADA al CCC, siendo éste quien se encargará de enviarla al elemento de campo correspondiente.

2. Posibilidad de añadir procesos gestores de alarmas / eventos.

   El CCC debe ser capaz de gestionar procesos tipo alarmas / eventos, o aplicaciones de control locales en los nodos que se requiera.

3. Posibilidad de adquisición de datos de entradas / salidas locales.

   Cuando se necesite gestionar entradas / salidas en un nodo el CCC debe ser capaz de gestionarlas directamente por un puerto dedicado a tal fin.

4. Ser considerado como un elemento de la misma familia por los dispositivos y redes de campo conectados a él.

   Al tener implementados los protocolos de comunicación de los elementos de campo conectados a él, el CCC será visto como otro elemento de la familia dado de alta por cada una de las redes que conforman el sistema de control. Los elementos que conforman las redes lo identificarán como un nodo más de su misma familia.

5. Posibilidad de dar de alta procesos de gestión de nuevos dispositivos a integrar.

   Muchas veces será necesario añadir nuevos elementos a integrar (por crecimiento del sistema o por nuevos requerimientos), por lo tanto el sistema debe ser capaz de permitir añadir nuevos dispositivos de manera simple.

6. Tener posibilidad de conexión a grandes redes de comunicación.

   Entre las prestaciones para conexión a los sistemas de comunicación superiores, el CCC dispondrá opcionalmente de puertos que permitan la conexión directa a las redes típicas de intercambio de grandes paquetes de información, por ejemplo LAN-ETHERNET. Para la gestión de estas redes es necesario incorporar los drivers correspondientes.

7. Capacidad de almacenamiento de datos.

   El CCC deberá contar con un dispositivo físico que permita el almacenamiento temporal de datos para su posterior tratamiento (por ejemplo archivos históricos). Este dispositivo físico no deberá tener partes móviles por las condiciones de los entornos industriales a nivel bajo. Ejemplos de este tipo de dispositivos son: tarjetas PCMCIA, discos de estado sólido, etc.

8. Permitir el análisis de protocolos de comunicación.

   El CCC deberá permitir analizar los protocolos de las redes de campo que integra, pues también podrá ser conectado a ellos sin ser dado de alta. En este caso tendrá por misión visualizar todas las transferencias de información en la red. Esta característica hace que el Concentrador facilite su implantación por etapas y con pérdidas de tiempo mínimas.

5.6 Topología de una red de CCCs.

Cuando la aplicación de integración, por cantidad o distribución espacial de los elementos de campo, deba ser realizada por más de un CCC, se deberá designar uno que se comporte como el CCCP. Del análisis realizado en los capítulos 3 y 4, se obtuvo entre otras conclusiones que la topología más eficiente utilizada por los buses de campo de alto y bajo nivel, es la de distribución de los elementos formando redes tipo árbol. En nuestro caso particular, y dado que utilizaremos para la comunicación entre CCCs un estándar de este tipo, la topología adoptada será también tipo árbol, por las ventajas ya vistas en el capitulo 3. En la Figura 45 se muestra una distribución típica de una red de CCCs.

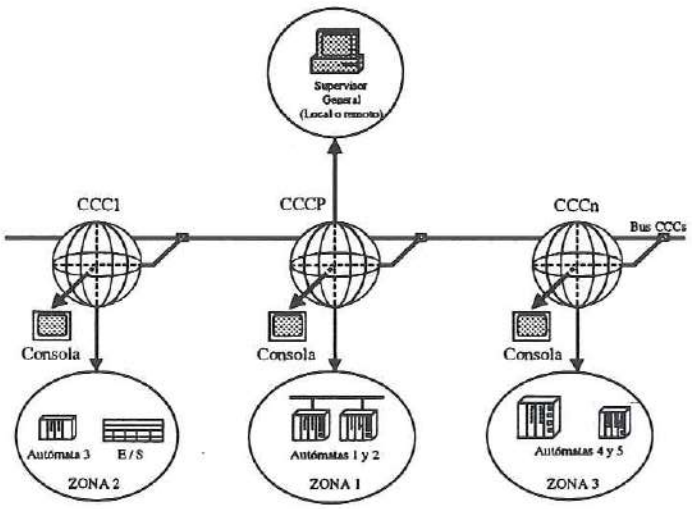


Figura 45: Ejemplo de red de CCCs.

En este tipo de red el CCCP tendrá la tabla general de objetos de comunicación y será el que se comunique con los niveles superiores mediante los "procesos de atención a supervisor" indicados en la Figura 44.

5.7 Selección del protocolo entre CCCs.

Para seleccionar el protocolo utilizado entre CCCs se ha considerado que éste debe presentar las características propias de un protocolo de un entorno industrial, del tipo que se plantea integrar, y que enumeramos a continuación:

1. Debe garantizar los tiempos de retardo de los mensajes en el bus.
2. El protocolo debe permitir la consistencia temporal de las variables que lo requieran.
3. Debe permitir la conexión y desconexión de nodos bajo tensión.
4. Debe ser un protocolo abierto.
5. Debe existir el hardware necesario para su implantación.
6. La conexión física entre nodos debe ser simple y flexible.
7. Las tramas de datos no deben ser largas.
8. El hardware debe tener precios bajos.

En este tipo de entorno se podría utilizar cualquiera de los protocolos utilizados para buses industriales, tanto los de alto nivel – que pueden transferir un número de datos considerable, más de 100 bytes según la definición dada por ISA para *fieldbuses*, (por ejemplo FIP, PROFIBUS, etc.) - como los buses de campo de más bajo nivel, como por ejemplo Bitbus, o cualquiera de los que utiliza CAN, por ejemplo SDS, DeviceNet, etc.

Puesto que con la metodología propuesta se integrarán los sistemas más bajos de la pirámide de automatización (por ejemplo autómatas, E/S, pequeñas redes de dispositivos, etc.), la cantidad de datos que manejarán los CCC será reducida, por ser en su mayoría datos tipo bit. Por ejemplo, un autómata de clase media de amplia utilización, como el S7-200 de Simatic, el SIEMENS SL5, MODICON TSX micro, etc. pueden tener en promedio 320 E/S digitales y 32 analógicas de 8 bits. Si se requiere la transferencia de todos estos datos hacen falta sólo 72 bytes. En los casos típicos analizados, los cuales se describen brevemente en las aplicaciones, no se supera este límite en ningún momento.

Teniendo en cuenta las consideraciones anteriores, analizaremos dos de los estándares más utilizados, a nivel de las comunicaciones de campo, como posibles estándares para el bus entre CCCs. Se analizarán WorldFIP y CAN por ser de los más utilizados en este tipo de entornos [Pint94].

5.7.1 Análisis de las prestaciones para los CCCs utilizando WorldFIP.

Para analizar la eficiencia de WorldFIP, como protocolo para las redes de CCCs, se puede recurrir al estudio de la codificación de sus tramas [Azev96]. En resumen todas las transacciones WorldFIP son realizadas en el intercambio de dos tramas: una trama llamada ID_DAT seguida por una trama RP_DAT.

La trama RP_DAT aparece dentro de un tiempo determinado. A este tiempo se le denomina tiempo de ida y vuelta (*TR turnaround time*), y es el tiempo empleado entre el final de la

recepción de una trama y el inicio de la transmisión de la siguiente. En la Figura 46 se muestra esta secuencia de tramas y el tiempo de ida y vuelta (*TR*).

El tiempo TR está definido en la parte de planificación de la red del estándar como $10 \times TMAC \leq TR \leq 70 \times TMAC$. Donde *TMAC* es el tiempo necesario para transmitir un bit del protocolo sobre el medio físico. En la Tabla 19 se muestra el valor de *TR* a distintas velocidades de transmisión.

Figura 46: Secuencia de tramas en WorldFIP

Velocidad de transmisión	*TR* mínimo= 10 *TMAC*	*TR* máximo= 70 *TMAC*
31.25 kb/s	320 µs	22.4 µs
1 Mb/s	10 µs	70 µs
2.5 Mb/s	4 µs	28 µs

Tabla 19: Tiempo *TR* en una red WorldFIP.

El valor de *TR* es uno de los principales parámetros para calcular la eficiencia de una red de este tipo.

De acuerdo a la especificación [Azev96] las tres capas OSI en que está especificado WorldFIP, agregan un total de 64 bits, para cualquier longitud de la información de usuario. La Figura 47 muestra los bits agregados en cada uno de los tres niveles OSI considerados en el estándar WorldFIP.

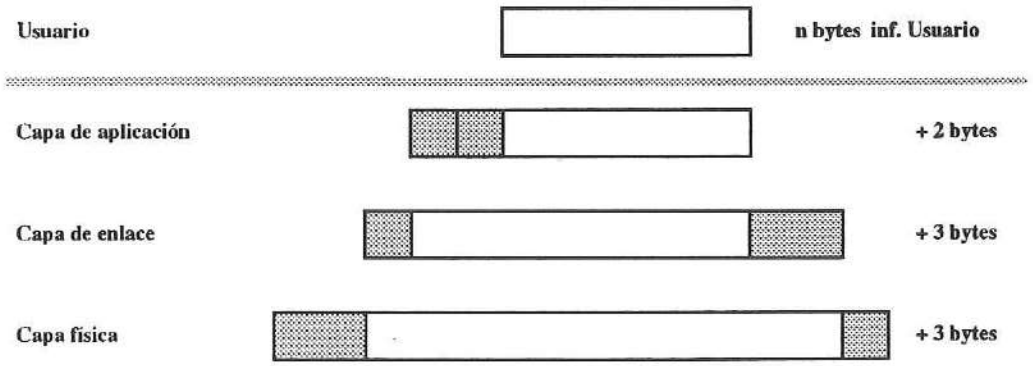


Figura 47: Bytes agregados en las capas OSI de WorldFIP

La realización de una transacción completa incluye los siguientes tiempos:

1. Transmisión de una trama de pregunta: $64 \times TMAC$ (constante)
2. Tiempo de ida y vuelta: *TR* donde ($TR \geq 10 \times TMAC$ y $TR \leq 70 \times TMAC$)

3. Transmisión de una trama de respuesta: $64 \times TMAC + n \times 8 \times TMAC$ (n = número de bytes de usuario)
4. Tiempo de ida y vuelta, igual que antes: TR donde ($TR \geq 10 \times TMAC$ y $TR \leq 70 \times TMAC$)

Podemos calcular la eficiencia como el cociente entre el tiempo empleado en transmitir la información del usuario y el tiempo total de la transacción, tal como muestra la Ecuación 1.

$$\eta_{FIP} = \frac{n \times 8 \times TMAC}{64 \times TMAC + TR \times TMAC + 64 \times TMAC + n \times 8 \times TMAC + TR \times TMAC}$$ Ecuación 1

La Tabla 20 muestra la eficiencia de WordlFIP para distintos valores de bytes de datos efectivos contenidos en la trama, basada en un TR = 10, y la Tabla 21 para TR = 70. Se muestra también el ancho de banda efectivo de utilización del bus para dos velocidades de transmisión 1 Mb/s y 2.5 Mb/s

Bytes de datos en la trama (n)	Eficiencia (η)	1 Mb/s	2.5 Mb/s
1	5.12 %	51.28 kb/s	128.21 kb/s
2	9.75 %	97.56 kb/s	243.90 kb/s
4	17.77 %	177.78 kb/s	444.44 kb/s
8	30.18 %	301.89 kb/s	754.72 kb/s
16	46.37 %	463.77 kb/s	1159.42 kb/s
32	63.36 %	633.66 kb/s	1584.16 kb/s
64	77.57 %	775.76 kb/s	1939.39 kb/s
128	87.37 %	873.72 kb/s	2184.30 kb/s

Tabla 20: Eficiencia y ancho de banda utilizado en WorldFIP (*TR*=10 *TMAC*)

Bytes de datos en la trama (n)	Eficiencia (η)	1 Mb/s	2.5 Mb/s
1	2.89 %	28.99 kb/s	72.46 kb/s
2	5.63 %	56.34 kb/s	140.85 kb/s
4	10.66 %	106.67 kb/s	266.67 kb/s
8	19.27 %	192.77 kb/s	481.93 kb/s
16	32.32 %	323.23 kb/s	808.08 kb/s
32	48.85 %	485.55 kb/s	1221.37 kb/s
64	65.64 %	656.41 kb/s	1641.03 kb/s
128	79.25 %	792.57 kb/s	1981.42 kb/s

Tabla 21: Eficiencia y ancho de banda utilizado en WorldFIP (TR=70 *TMAC*)

De estas tablas podemos deducir que cuando los datos del usuario son de tamaño reducido el tiempo TR es un factor predominante. Para el caso de 8 bytes de datos, puesto en negritas en las tablas, tenemos que la eficiencia es 30.18% en el mejor de los casos, es decir para (*TR*=10 *TMAC*). La situación es peor si el (*TR*=70 *TMAC*) donde la eficiencia disminuye al valor 19.27%.

5.7.2 Análisis de las prestaciones para los CCCs de un bus CAN.

La Figura 40 muestra la forma de las tramas, tramas con datos, en un bus CAN para la versión de identificador estándar y extendido, de 11 y 29 bits respectivamente. Las tramas para petición remota de datos son similares pero hay que recordar que no contienen datos. La Tabla 22 muestra las partes de las tramas de datos, según [CAN2.0], para la versión de identificador estándar a 11 bits, y la Tabla 23 para la versión extendida a 29 bits. Hay que tener en cuenta que además de esta tabla se tienen que considerar los bits de relleno. Estos bits de relleno, según la especificación, son añadidos por el transmisor cuando aparecen 5 bits consecutivos con igual nivel (*bit stuffing*), y son desechados por el receptor. El método de los bits de relleno empieza con el inicio de la trama y termina con la secuencia de CRC. Ésto hace que la longitud de las tramas CAN dependa del valor del dato.

Significado	No. de bits	Comentarios
Inicio de trama (SOF)	1	No se cuenta para cálculo de bits de relleno
Campo de arbitraje de los mensajes (identificador)	11 + 1	11 de identificador Más 1 que indica si es una transmisión de petición remota
Campo de control	2 + 4	2 bits de control. 1 de tipo de identificador y otro reservado + 4 bits para codificar la longitud del campo de datos (0 a 8 byte)
Campo de datos	0 a 64 = $n\times8$	0 a 8 Bytes. (n = número de bytes de datos)
Campo de CR	15 + 1	Secuencia CRC + delimitador de CRC
Bits para el cálculo de los bits de relleno = *Nbt*	34 + $n\times8$	No se cuenta el bit de inicio de trama
Campo de reconocimiento	2	Bit de ACK + delimitador de ACK
Fin de trama (EOF)	7	Bits recesivos
Total de bits en la trama	44 + $n\times8$	Identificador estándar
Bits de reposo del bus	3	Luego del final de cada una de las tramas

Tabla 22: Estructura de una trama de datos estándar en CAN.

Significado	No. de bits	Comentarios
Inicio de trama (SOF)	1	No se cuenta para cálculo de bits de relleno
Campo de arbitraje de los mensajes	11 + 2 18 + 1	11 de identificador + 2 de tipo identificador 18 de identificador + 1 para indicar transmisión remota
Campo de control	2 + 4	2 bits de control reservados para uso futuro + 4 bits para codificar la longitud del campo de datos (0 a 8 byte)
Campo de datos	0 a 64 = $n\times8$	0 a 8 Bytes. (n = número de bytes de datos)
Campo de CRC	15 + 1	Secuencia CRC + delimitador de CRC
Bits para el cálculo de los bits de relleno = *Nbt*	54 + $n\times8$	No se cuenta el bit de inicio de trama
Campo de reconocimiento	2	*Slot* de ACK + delimitador de ACK
Fin de trama (EOF)	7	Bits recesivos
Total de bits en trama	64 + $n\times8$	Identificador extendido
Bits de reposo	3	Luego del final de cada una de las tramas

Tabla 23: Estructura de una trama de datos extendida en CAN.

Debido al método de relleno el tiempo empleado en trasmitir una trama de datos - t_T -, tiene un valor mínimo y un máximo. El número máximo de bits de relleno debe ser calculado para el peor caso. Aparentemente el peor caso sería el de un mensaje con todos los bits a uno o a cero, como sigue:

1111 1111 1111 1111 1111 1...

En este caso particular, o en el complementario en el que todos los bits están a cero, el método de relleno produce la secuencia:

1111 **1**011 111**0** 1111 **1**011 111**0** 1...

La expresión matemática, para calcular el número de bits de relleno - representados por los números en negritas -, está expresada por la Ecuación 2.

$$Nbr = \left[\frac{Nbt}{5}\right] \qquad \text{Ecuación 2}$$

En donde:

Nbr : Número de bits de relleno.
Nbt : Número de bits totales para cálculo del relleno.

Pero el peor caso, que no es el expuesto antes, se presenta en la siguiente secuencia de bits, y en su complementaria:

1111 1000 0111 1000 0111 1...

En este caso el método de relleno genera la secuencia:

1111 **1**000 00111 11**0**0 000**1** 1111 **0**...

De donde podemos deducir que la expresión que indica el peor caso de bits de relleno, para una secuencia como la indicada antes, está dada por la Ecuación 3.

$$Nbr = \left[\frac{Nbt-1}{4}\right] \qquad \text{Ecuación 3}$$

Con el valor del número de bits de relleno, dado por la Ecuación 3 , podemos calcular cual es la eficiencia de CAN para transmitir un mensaje de n bytes. La Ecuación 4 y la Ecuación 5 indican estos valores para mensaje estándar y extendido respectivamente.

$$\eta = \frac{n \times 8}{44 + 3 + 8 \times n + \left[\frac{33 + 8 \times n}{4}\right]} \qquad \text{Ecuación 4}$$

$$\eta = \frac{n \times 8}{64 + 3 + 8 \times n + \left\lceil \frac{53 + 8 \times n}{4} \right\rceil}$$ Ecuación 5

El valor de la eficiencia para transmitir mensajes, con longitudes de datos entre 1 y 8 bytes, se muestra en la Tabla 24 para mensajes con identificador estándar y extendido.

Bytes de datos en la trama (*n*)	Identificador estándar.	Identificador extendido.
1	12.31 %	8.88 %
2	21.33 %	16.00 %
3	28.23 %	21.82 %
4	33.68 %	26.67 %
5	38.10 %	30.77 %
6	41.74 %	34.28 %
7	44.80 %	37.33 %
8	47.40 %	40.00 %

Tabla 24: Eficiencia de CAN para transmitir mensajes de datos.

Aunque los mensajes de petición remota son muy esporádicos, también analizamos a continuación la eficiencia de CAN para transmitirlos. Según el estándar [CAN2.0] las partes de una trama remota se muestran en la Tabla 25 y Tabla 26, para mensajes de petición remota con identificador estándar y extendido respectivamente. Luego de realizada la petición remota el nodo receptor de la petición responde con una trama de datos tal como las indicadas en la Tabla 22 y en la Tabla 23 según sea el caso.

Significado	No. de bits	Comentarios
Inicio de trama (SOF)	1	
Campo de arbitraje de los mensajes (identificador)	11 + 1	11 de identificador Más 1 que indica si es una transmisión de petición remota
Campo de control	2 + 4	2 bits de control 1 de tipo de identificador y otro reservado + 4 bits para codificar la longitud del campo de datos (0 a 8 byte)
Campo de CRC	15 + 1	Secuencia CRC + delimitador de CRC
Bits para el cálculo de los bits de relleno = *Nbt*	34	No se cuenta el bit de inicio de trama
Campo de reconocimiento	2	*Slot* de ACK + delimitador de ACK
Fin de trama	7	Bits recesivos
Total de bits en la trama	44	Identificador estándar
Bits de reposo del bus	3	Luego del final de cada una de las tramas

Tabla 25: Estructura de una trama remota con identificador estándar.

S: ficado	No. de bits	Comentarios
Inicio de tra . (SOF)	1	
Campo de arbitraje de los mensajes	11 + 2 18 + 1	11 de identificador + 2 de tipo identificador 18 de identificador + 1 para indicar transmisión remota
Campo de control	2 + 4	2 bits de control reservados para uso futuro. + 4 bits para codificar la longitud del campo de datos (0 a 8 byte)
Campo de CRC	15 + 1	Secuencia CRC + delimitador de CRC
Bits para el cálculo de los bits de relleno = *Nbt*	54	No se cuenta el bit de inicio de trama
Campo de reconocimiento	2	*Slot* de ACK + delimitador de ACK
Fin de trama	7	Bits recesivos.
Total de bits en la trama	64	Identificador estándar
Bits de reposo del bus	3	Luego del final de cada una de las tramas.

Tabla 26: Estructura de una trama remota con identificador extendido.

Con los datos de la Tabla 25 y de la Tabla 26 respectivamente, podemos calcular la eficiencia de CAN cuando se realiza una petición remota de datos. En estos casos la expresión de la eficiencia está dada por la Ecuación 6 , para identificador estándar; y por la Ecuación 7 para identificador extendido. Los resultados se muestran en la Tabla 27.

$$\eta = \frac{n \times 8}{99 + 8 \times n + \left\lfloor \frac{33 + 8 \times n}{4} \right\rfloor}$$ Ecuación 6

$$\eta = \frac{n \times 8}{144 + 8 \times n + \left\lfloor \frac{53 + 8 \times n}{4} \right\rfloor}$$ Ecuación 7

Bytes de datos En la trama (*n*)	Identificador estándar.	Identificador extendido.
1	6.83 %	4.80 %
2	12.60 %	9.04 %
3	17.52 %	12.83 %
4	21.77 %	16.24 %
5	25.48 %	19.32 %
6	28.74 %	22.12 %
7	31.60 %	24.67 %
8	34.22 %	27.00 %

Tabla 27: Eficiencia de CAN para transmitir mensajes de petición remota.

Para completar el análisis de CAN en las tablas siguientes se comparan los tiempos empleados para transmitir mensajes, normales y de petición remota, para tramas con identificador de 11 y 29 bits respectivamente. Los tres casos han sido evaluados para longitudes de datos entre 0 y 8 bytes; uno a velocidad de transmisión de 125 kb/s (Tabla 28), otro a 500 kb/s(Tabla 29), y el último a la velocidad máxima permitida por el estándar, 1Mb/s (Tabla 30). Se indica también el porcentaje de tiempo adicional que requieren las tramas de mensajes con identificador de 29 bits frente a las de 11 bits. Se hace esta comparación para dos casos extremos, el caso sin bits de relleno y el caso con número máximo de bits de relleno.

	Identificador.	Longitud del mensaje de datos (Bytes)								
		0	1	2	3	4	5	6	7	8
Mensaje normal Sin bits de relleno	11-bit-ID	0.55	0.65	0.75	0.85	0.95	1.05	1.15	1.25	1.35
	29-bit-ID	0.8	0.9	1	1.1	1.2	1.3	1.4	1.5	1.6
Diferencia en %		45.45%	38.46%	33.33%	29.41%	26.32%	23.8%	21.74%	20%	18.52%
Mensaje normal Bits de relleno máx	11-bit-ID	0.69	0.81	0.94	1.06	1.19	1.31	1.44	1.56	1.69
	29-bit-ID	0.96	1.09	1.21	1.34	1.46	1.59	1.71	1.84	1.96
Diferencia en %		40%	33.85%	29.33%	25.88%	23.15%	20.95%	19.13%	17.6%	16.30%
Mensaje remoto Sin bits de relleno	11-bit-ID	1.14	1.24	1.34	1.44	1.54	1.64	1.74	1.84	1.94
	29-bit-ID	1.64	1.74	1.84	1.94	2.04	2.14	2.24	2.34	2.44
Diferencia en %		43.96%	40.4%	37.38%	34.78%	32.52%	30.53%	28.78%	27.21%	25.81%
Mensaje remoto Bits de relleno máx	11-bit-ID	1.34	1.46	1.59	1.71	1.84	1.96	2.09	2.21	2.34
	29-bit-ID	1.96	2.09	2.21	2.34	2.46	2.59	2.71	2.84	2.96
Diferencia en %		46.73%	42.74%	39.37%	36.5%	34%	31.85%	29.94%	28.25%	26.74%

Tabla 28: Tiempo de envío de mensajes CAN (en ms) a 125 kb/s

	Identificador.	Longitud del mensaje de datos (Bytes)								
		0	1	2	3	4	5	6	7	8
Mensaje normal Sin bits de relleno	11-bit-ID	88	104	120	136	152	168	184	200	216
	29-bit-ID	128	144	160	176	192	204	224	240	256
Diferencia en %		45.45%	38.46%	33.33%	29.41%	26.32%	23.8%	21.74%	20%	18.52%
Mensaje normal Bits de relleno máx	11-bit-ID	110	130	150	170	190	210	230	250	270
	29-bit-ID	154	174	194	214	234	254	274	294	314
Diferencia en %		40%	33.85%	29.33%	25.88%	23.15%	20.95%	19.13%	17.6%	16.30%
Mensaje remoto Sin bits de relleno	11-bit-ID	182	198	214	230	246	262	278	294	310
	29-bit-ID	262	278	294	310	326	342	358	374	390
Diferencia en %		43.96%	40.4%	37.38%	34.78%	32.52%	30.53%	28.78%	27.21%	25.81%
Mensaje remoto Bits de relleno máx	11-bit-ID	214	234	254	274	294	314	334	354	374
	29-bit-ID	314	334	354	374	394	414	434	454	474
Diferencia en %		46.73%	42.74%	39.37%	36.5%	34%	31.85%	29.94%	28.25%	26.74%

Tabla 29: Tiempo de envío de mensajes CAN (en µs) a 500 kb/s

	Identificador.	Longitud del mensaje de datos (Bytes)								
		0	1	2	3	4	5	6	7	8
Mensaje normal Sin bits de relleno	11-bit-ID	44	52	60	68	76	84	92	100	108
	29-bit-ID	64	72	80	88	96	104	112	120	128
Diferencia en %		45.45%	38.46%	33.33%	29.41%	26.32%	23.8%	21.74%	20%	18.52%
Mensaje normal Bits de relleno máx	11-bit-ID	55	65	75	85	95	105	115	125	135
	29-bit-ID	77	87	97	107	117	127	137	147	157
Diferencia en %		40%	33.85%	29.33%	25.88%	23.15%	20.95%	19.13%	17.6%	16.30%
Mensaje remoto Sin bits de relleno	11-bit-ID	91	99	107	115	123	131	139	147	155
	29-bit-ID	131	139	147	155	163	171	179	187	195
Diferencia en %		43.96%	40.4%	37.38%	34.78%	32.52%	30.53%	28.78%	27.21%	25.81%
Mensaje remoto Bits de relleno máx	11-bit-ID	107	117	127	137	147	157	167	177	187
	29-bit-ID	157	167	177	187	197	207	217	227	237
Diferencia en %		46.73%	42.74%	39.37%	36.5%	34%	31.85%	29.94%	28.25%	26.74%

Tabla 30: Tiempo de envío de mensajes CAN (en µs) a 1 Mb/s

5.7.3 Conclusiones para la red de interconexión de CCCs.

Luego del análisis anterior podemos sacar las siguientes conclusiones:

1. WorldFIP es un bus eficiente para transferencias donde la cantidad de bytes de datos efectivos es elevada, tal como muestran la Tabla 20 y la Tabla 21. Dado que la presente tesis está orientada a la concentración y transferencia de datos, de elementos de campo, que ocupan pocos bytes, este bus no se adecua a nuestros requerimientos. Cabe resaltar también que el costo de los chips de interfaz de WorldFIP es al menos 5 veces mayor que el de los chips de CAN.

2. Del análisis de eficiencia realizado en 5.7, y recordando que la cantidad de datos que manejaremos será pequeña, podemos concluir que CAN es el que mejor se adapta a nuestros requerimientos para interconectar redes de CCCs.

3. Los tiempos máximos empleados para transmitir un mensaje en CAN, como por ejemplo los mostrados en la Tabla 29 y Tabla 30, más los retardos que origine el procesador *host*, que se comunica con el chip de interfaz, pueden ser garantizados como los tiempos de respuesta del objeto con la máxima prioridad en este tipo de bus. Este tiempo, a diferencia de otros buses, es el tiempo que tarda el objeto en ser transmitido entre dos nodos cualesquiera en el sistema. Ésto es debido a que CAN es un sistema multimaestro y por consiguiente el objeto no tiene que pasar a un maestro central, como se necesita en los sistemas de *polling*. Más aún, cada nodo puede obtener directamente la información de otro nodo, realizando una petición remota.

4. El filtrado de los mensajes, en una red CAN, es quizás una de las acciones que más influirá en el tiempo de retardo del software; ya que si no filtramos los mensajes que no van dirigidos a la CPU de un nodo, ésta será interrumpida constantemente, por mensajes que no deben recibir.

5. La parte más importante del tiempo de retardo, en una red CAN, es el tiempo empleado en transmitir una trama t_T. Este tiempo está determinado por la longitud de la trama n y la velocidad de transmisión de los datos. Los valores de la Tabla 28, Tabla 29 y Tabla 30 muestran que la influencia de un identificador extendido en el retardo es muy grande. Por ejemplo en una aplicación con longitudes de datos de hasta 4 bytes, significa que aumentamos el retardo un 30% respecto al uso de mensajes con identificador estándar. Hay que tener en cuenta también que este hecho influencia el tiempo de ciclo mínimo.

6. Utilizaremos mensajes con identificador estándar, pues, en la mayoría de aplicaciones industriales el uso de mensajes de este tipo es suficiente; ya que permite direccionar hasta 2032 objetos de comunicación diversos. En caso de ser necesario se puede utilizar la versión con identificadores extendidos, considerando que en este caso la longitud de la trama se extiende en 20 bits según la especificación de 2.0B CAN.

5.8 Tipos de transferencias entre nodos.

En el análisis anterior se ha demostrado que CAN es el que más se adecua a los requerimientos de la red de CCCs. Para las transferencias de los objetos de cada CCC necesitamos que el sistema de transmisión utilizado, basado en CAN, transfiera la mayor cantidad de bytes útiles de datos en cada conexión – pues el objetivo fundamental de la red de CCCs será la integración de todas las variables de los dispositivos -, para optimizar los tiempos de transferencia. Por lo tanto, como este estándar sólo define las dos primeras capas del modelo OSI, seguidamente se realiza una evaluación, como posibles redes de integración para los CCCs, de dos de las aplicaciones comerciales de capa 7, basadas en este estándar, que permiten el envío fragmentado de datos (caso de la tabla completa de cada nodo) en el caso de que no se pueda enviar toda la información en una sola conexión.

1. SDS:
   Esta red permite transferir mensajes fragmentados hasta en 63 partes. Este sería la forma de funcionamiento que podríamos adoptar en la red de CCCs. Para permitir esta forma de transmisión SDS utiliza cuatro bytes del campo de datos (de los ocho destinados para tal fin) para gestión del protocolo correspondiente. La división del campo de datos en este modo de funcionamiento se muestra en la Figura 48. Los bytes de datos sombreados son los utilizados por el protocolo.

	Bit							
Dato	7	6	5	4	3	2	1	0
0	Pregunta/respuesta/error		Fragmen. 1	Grupo de objetos (0 ... 31)				
1	Modificador (0 ... 255)							
2	Número de fragmento (0 ... 63)							
3	Total de bytes fragmentados (0 ... 255)							
4	Byte de datos 0							
5	Byte de datos 1							
6	Byte de datos 2							
7	Byte de datos 3							

Figura 48: División del campo de datos en SDS para mensajes fragmentados.

2. DeviceNet:
   Esta red está diseñada para integrar gran cantidad de dispositivos de campo aunque también permite transferencias de mensajes fragmentados utilizando sólo un byte del campo de datos para gestión del protocolo. La división del campo de identificador se realizada tal como indica la Figura 49 obteniéndose 4 grupos de mensajes.

Bits del identificador											Cantidad	Referencia
10	9	8	7	6	5	4	3	2	1	0		
0	Identificador Grupo 1				Identificador de acceso						1024	Grupo 1
1	0	Identificador de acceso						Identificador Grupo 2			512	Grupo 2
1	1	Identificador Grupo 3			Identificador de acceso						448	Grupo 3
1	1	1	1	1	Identificador Grupo 4						48	Reservado
1	1	1	1	1	1	1	X	X	X	X	No permitido	

Figura 49: División del campo de identificador en DeviceNet.

De estas redes la que mejor se podría adaptar a nuestros propósitos es la segunda, pero presenta, a parte de ocupar un byte para gestión de protocolo, el inconveniente de causar interrupciones en algunos nodos que no están involucrados en la transferencia. Este tipo de redes son adecuadas en la interconexión de gran cantidad de elementos, pero, aunque permiten la transferencia de mensajes fragmentados, no son tan eficientes en esta forma de funcionamiento.

Para definir los tipos de transferencias en la red CAN entre CCCs se propone el uso optimizado de los once bits del campo de identificación, para poder disponer del 100% del campo de datos, en una distribución como la que se muestra en la Tabla 31, y con interfaz a bus de tipo extendida (14 mensajes-objeto CAN de envío / recepción y 1 dedicado a la recepción con doble buffer).

Campo de control			Campo de identificación y operación								Referencia
ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0	Bits del identificador.
0	0	0	X	X	X	X	X	X	X	X	Reservado
0	0	1	Número del mensaje 0 – 255								Difusión, y mensj. con restricciones
0	1	0	Número del nodo 0 – 15				Tipo de operación				Mensajes de operación CCCP
0	1	1	Número del nodo 0 – 15				Núm. de fragmento 0 - 15				Tabla de CCCP a CCCn
1	0	0	Número del nodo 0 – 15				Tipo de operación				Mensajes de operación CCC
1	0	1	Número del nodo 0 – 15				Núm. de fragmento 0 - 15				Tabla de CCCn a CCCP
1	1	0	X	X	X	X	X	X	X	X	Reservado
1	1	1	X	X	X	X	X	X	X	X	Reservado
1	1	1	1	1	1	1	X	X	X	X	No permitido

Tabla 31: División de los bits del identificador.

Con esta distribución se podrán transferir:

- Todos los objetos que agrupan a las variables sin restricciones de tiempo: para esto se utilizará el mensaje-objeto 1 de la interfaz CAN, con un máximo de 16 fragmentos (cada uno con 8 bytes de datos) y 16 nodos. También por este mensaje-objeto CAN se transmitirán los mensajes de operación.

- Mensajes de difusión del CCCP: Transmitidos para algunos mensajes especiales que involucren a todos los nodos. Existe uno que será emitido a intervalos fijos de tiempo para indicar qué nodos se encuentran activos, y para sincronizar los relojes internos de los mismos. Estos mensajes son transmitidos por el mensaje - objeto número 2 de la interfaz a bus CAN,

- Mensajes con restricciones: se utilizarán los 12 mensajes-objetos libres en los nodos (del 3 al 14) para permitir la transmisión de mensajes con algún tipo de restricción, por ejemplo: mensajes que deban cumplir consistencia temporal, mensajes que deben ser recibidos al mismo tiempo por más de un nodo, mensajes cíclicos o con otro tipo de restricción de tiempo. El peor de los casos ocurrirá cuando todos los nodos tengan definidos 6 objetos de transmisión y 6 de recepción lo que hace un total de $NMC = 6 \times 16 = 96$, en donde NMC es el número de mensajes que deben cumplir consistencia temporal o restricciones de tiempo.

Los campos indicados en la Tabla 31 tienen la funcionalidad siguiente:

Campo de control (bits ID10, ID9 e ID8): Permite definir los diferentes tipos de mensajes que se pueden tener entre CCCs. En la Tabla 31 se indica el tipo de operaciones que pueden permitir estos tres bits del campo de control. La opción (1, 1, 1, 1, 1, 1, 1) no es posible ya que CAN especifica que no se deben asignar identificadores con los 7 bits más significativos a 1. También existen números de control, como el (0, 0, 0), el (1, 1, 0) y las combinaciones indicadas en la penúltima fila (que cumplen la restricción del estándar) que no se han utilizado y quedan reservados para futuras aplicaciones.

Campo de identificación y operación (bits ID7 a ID0): Este campo, de acuerdo a los valores del campo de control, tomará un solo valor o se dividirá en dos. En el caso de los mensajes con consistencia o de difusión del CCCP este campo toma los 8 bits de identificación con los que se pueden definir hasta 256 mensajes. El mensaje cero se emplea para los mensajes de difusión del CCCP por tener la más alta prioridad. Los otros 255 identificadores se utilizan para definir los mensajes entre nodos que deben tener consistencia temporal.

Cuando se trata de mensajes de envío de datos de los elementos que se están integrando, o bien de mensajes de operación, este campo se subdivide en dos. El primer campo, de 4 bits indica el número de nodo – por consiguiente permite direccionar hasta 16 nodos, pero reservando el cero para el CCCP -, y en el segundo campo, también de 4 bits, se indica el número de fragmento (paquete de datos) en curso, o bien el número de operación que se está realizando, respectivamente.

De esta manera la tabla de objetos de un CCC deberá tener una estructura como la que se indica en la Figura 50. En el CCCP esta estructura contendrá todas las tablas de los nodos de la red.

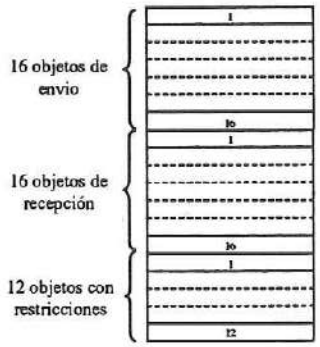


Figura 50: Estructura de la tabla de objetos de un CCC.

5.8.1 Mensajes de transferencia sin restricciones.

Los CCCs deberán agrupar las variables, sin restricciones, que enviarán y recibirán de los dispositivos a integrar para su transmisión más eficiente en la tabla de objetos, que luego serán transformados, por los procesos de envío y recepción, en mensajes-objeto CAN para su transmisión al bus.

De la Tabla 31 podemos deducir que las cantidades máximas de datos que cada CCC podrá intercambiar con el CCCP, para transferencia de las variables sin restricciones serán:

Variables	Exportar	Importar	Total Intercambiado.
Tipo bit.	1024	1024	2048
Tipo Byte.	128	128	256
Tipo doble Byte	64	64	128

Tabla 32: Cantidad de variables sin restricciones integradas por cada nodo.

Se puede transferir cualquier combinación que no supere, en exportación o importación, los 128 bytes de capacidad. Se ha seleccionado esta cantidad pues, como se indicó en 5.7, el número de variables integradas en un sistema como el estudiado raramente superará este valor. Las variables exportadas por los nodos se envían haciendo uso del mensaje objeto numero 1 del chip de interfaz; y las importadas, del CCCP, por el objeto 15 que se define para esta función. Este objeto de recepción tiene un doble buffer de memoria, por lo que puede recibir dos mensajes uno tras otro sin perderlos.

En el caso de que los requerimientos del sistema supere la cantidad de variables que se pueden integrar, o se supere el número máximo de nodos, entonces se podría utilizar la versión extendida de identificador a 29 bits. De esta manera el campo de identificación y operación se extendería de 8 bits a 26, pudiéndose direccionar hasta $2^{13} = 8192$ nodos con igual cantidad de conexiones para la transferencia de la tabla de objetos. La división de los bits del identificador se realizaría de la forma indicada en la Tabla 33. El total de variables que teóricamente se podría exportar en este caso se muestra en la Tabla 34.

Campo de control			Campo de identificación y operación						Referencia
ID28	ID27	ID26	ID25	...	ID13	ID12	...	ID0	Bits del identificador.
0	0	0	X	...	X	X	...	X	Reservado
0	0	1	Número del mensaje 0 – 67108863						Difusión, y mensj. con restricciones
0	1	0	N. nodo 0 – 8192			Tipo de operación			Mensajes de operación CCCP
0	1	1	N. nodo 0 – 8192			N. fragmento 0 - 8192			Tabla de CCCP a CCCn
1	0	0	N. nodo 0 – 8192			Tipo de operación			Mensajes de operación CCC
1	0	1	N. nodo 0 – 8192			N. fragmento 0 - 8192			Tabla de CCCn a CCCP
1	1	0	X	...	X	X	...	X	Reservado
1	1	1	X	...	X	X	...	X	Reservado
1	1	1	1	1	1	1	...		No permitido

Tabla 33: División del identificador para el caso de formato extendido.

Variables	Exportar	Importar	Total Intercambiado.
Tipo bit.	524288	524288	1048576
Tipo Byte.	65536	65536	131072
Tipo doble Byte	32768	32768	65536

Tabla Cantidad de variables sin restricciones integradas por cada nodo con identificador extendido.

5.8.2 Mensajes de operación.

Cada nodo implementa un sistema de mensajería con el CCCP. Se podrían definir hasta 16 grupos distintos de operaciones, y a su vez cada grupo podrá tener definidas hasta 256 operaciones distintas. Para definir estas operaciones sólo se hará uso de un byte de datos del mensaje de operación, que si se quiere se puede extender hasta 2^{64} operaciones utilizando los 8 bytes de datos. Los grupos de operación definidos se indican en la Tabla 35.

Identificador				Descripción
ID3	ID2	ID1	ID0	
0	0	0	0	Indicación de nodo activo
0	0	0	1	Indicación de alarma
0	0	1	0	Secuencia de apagado
0	0	1	1	Petición de repetición
0	1	0	0	Reconexión
0	1	0	1	Reset
0	1	1	0	Desconexión de red
0	1	1	1	Cambio de variables
1	0	0	0	Reservado
1	0	0	1	Reservado
1	0	1	0	Reservado
1	0	1	1	Reservado
1	1	0	0	Reservado
1	1	0	1	Reservado
1	1	1	0	Reservado
1	1	1	1	Reservado

Tabla 35: Mensajes de operación de un CCC.

Indicación de nodo activo: Se envía un mensaje al CCCP, cada 500 ms para indicar que no ha ocurrido ningún cambio de las variables locales, pero que el nodo está activo. Esta comunicación sirve al CCCP para poner a cero el temporizador, que arranca desde que llegó el último mensaje del nodo, y que estará configurado para 500 ms. Si no llega la indicación de nodo activo el CCCP indica una anomalía de funcionamiento en el nodo que no ha informado.
Indicación de alarma: Envía un mensaje indicando que una alarma, definida en la configuración, se ha activado. El número de la alarma se indica en un byte de datos del mensaje.
Secuencia de apagado: Indica al CCCP que el nodo saldrá de servicio y que lo elimine de su lista de nodos activos sin originar mensaje de alarma.
Petición de repetición: Si al hacer la verificación de los mensajes recibidos, se determina que no han llegado todos, el nodo hace una petición de repetición del mensaje.
Reconexión: Cuando un nodo requiere darse de alta nuevamente en la red, envía esta petición al CCCP y mientras este no le autorice no podrá enviar sus datos a la red.
Reset: Esta operación puede ser realizada local o remotamente desde el CCCP originado el reset del nodo.
Desconexión de red: El CCCP podrá ordenar con esta operación que un nodo salga de la red hasta nuevo aviso.
Cambio de variables: Indica que se han eliminado algunas variables definidas o que se han definido variables en el nodo. Ésto origina una reconfiguración, en línea, por parte del CCCP.

Con estas operaciones definidas, obtenemos la funcionalidad deseada del sistema.

5.8.3 Mensajes de difusión del CCCP y mensajes, entre nodos, que presenten restricciones.

El nodo CCCP tiene definido un mensaje de difusión para indicar algunas operaciones comunes a todos los nodos, como por ejemplo fin de configuración remota, reset global, cambio de velocidad de bus, paro de emergencia, sincronización de relojes, etc. El tipo de operación se define, al igual que para la transferencia de mensajes de operación, en un solo byte (hasta 256); pero pudiéndose extender también hasta 2^{64}. En la Tabla 31, de los 256 identificadores posibles para este grupo, utilizamos el número cero para este mensaje, es decir el más prioritario. Los demás nodos reciben el mensaje de difusión del CCCP en el objeto número 2, al cual se le asigna exactamente el número de identificación indicado.

Cuando algunas variables tengan restricciones particulares, por ejemplo envío cíclico, consistencia temporal, etc., éstas también utilizarán identificadores en el rango antes mencionado. La prioridad de las mismas será menor que las de los mensajes de emisión del CCCP.

La configuración de este tipo de variables la realizará automáticamente el CCCP en el momento de la configuración remota del sistema. Él es el que se encarga de asignarles los identificadores a los mensajes y de indicar a los nodos con cuales de ellos deben configurar sus 12 objetos libres.

5.9 Pasos a seguir para la implantación de la metodología propuesta.

En este apartado se describen los pasos a seguir para lograr la integración utilizando la metodología propuesta. La secuencia de los pasos se muestra en la Tabla 36.

Paso	Descripción
1	Descripción de los dispositivos a integrar
2	Entrada de datos en cada uno de los nodos
3	Configuración remota
4	Funcionamiento normal

Tabla 36: Pasos a seguir.

En los apartados siguientes se describen cada uno de ellos.

5.9.1 Descripción de los dispositivos a integrar.

En primer lugar es necesario detallar los dispositivos a integrar incluyendo el número y tipo de variables que deben intercambiar. Los datos deben ser organizados en tablas, como la que se indica en la Tabla 37. El número de tablas de este tipo será igual al número de zonas de integración que definamos. Una zona será definida tomando como referencia las distancias que aceptan los puertos de los dispositivos a integrar. Con el dato de la distancia de cada dispositivo podemos decidir qué elementos pueden formar una zona de integración e ir conectados al mismo CCC. En esta tabla se indica (mediante un número del 1 al 256) el driver utilizado para comunicarse con el dispositivo a integrar.

ZONA n									
Tipo de dispositivo	Conexión al CCC	Driver	# Variables a exportar			# Variables a importar			Restricciones de tiempo
			Bit	Byte	Long	Bit	Byte	Long	
PLC AB502	RS-232	1	128	8	0	10	1	0	No
PLC AB503	RS-232	2	64	2	0	8	0	0	Si
Salida discreta	Bit de entrada	3	1	0	0	0	0	0	No
Entrada discreta	Bit de salida	3	0	0	0	1	0	0	No

Tabla 37: Tabla de dispositivos por zonas.

5.9.2 Entrada y organización de datos en cada uno de los CCC.

En cada CCC se deben introducir las variables correspondientes, según el formato mostrado en el ejemplo de la Tabla 38.

Los dos primeros campos indican la referencia numérica y textual del dato en el nodo. El campo "tipo de variable" indica si la variable es tipo bit (b) tipo byte (B) o tipo doble byte (L). En los dos campos siguientes se indica el driver utilizado y el número de puerto por donde se realiza el intercambio.

Nº de referencia de la variable	Descripción	Tipo de variable	Driver	Puerto	Envío / recepción	Nodo(s) de destino Nodo de origen	Restric-ciones
1	Foco1	b	3	1	Envío	P	0
2	Foco3	b	1	2	Envío	P, 2	0
5	Motor5	B	1	3	Envío	5, 6, 7	t
14	Motor6	B	2	3	Recepción	15	0
15	Pulsad1	b	3	2	Recepción	3	0
16	Conmut0	L	1	1	Envío	15	2t
123	Bomba1	b	1	3	Envío	12, 2	0
124	Ventilador0	b	2	1	Envío	P	t
125	Luz_verde	b	1	2	Envío	15, 1, 5	0
126	Cizalla	L	2	3	Envío	P	3t
128	Dosificador	L	2	2	Envío	P, 13, 11	0
150	Torno	L	2	2	Recepción	P	3t

Tabla 38: Tabla de variables locales a introducir en un nodo CCC.

En el campo envío / recepción se indica si la variable es enviada del nodo o recibida en el mismo. Seguidamente se indica el campo de nodo(s) de destino, u origen, de la variable específica (cada uno de los 16 bits de este campo indicará si el nodo correspondiente se encuentra activo). El último campo indica si la variable tiene algún tipo de restricción (por ejemplo cíclica –indicando el tiempo de ciclo (como múltiplo de un tiempo "t" que dependerá de la velocidad del bus) -, o restricciones de consistencia) si no presenta ninguna restricción entonces la variable será enviada en el menor tiempo que el sistema lo permita. Este tiempo será calculado oportunamente por el CCCP. En una red de CCCs se enviará esta tabla al CCCP, en el momento de que éste realiza la configuración del sistema. Esta tabla es almacenada, localmente, y en el CCCP, en una estructura de memoria como indica la Figura 51 donde también se indica el significado de cada campo.

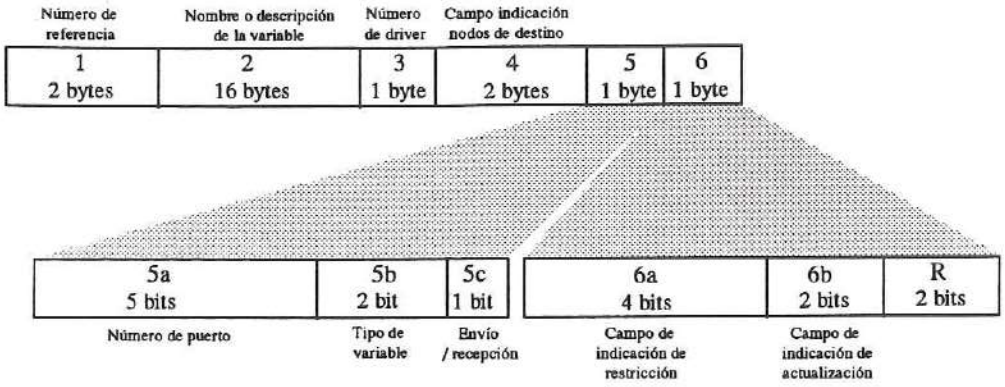


Figura 51: Estructura en memoria de la tabla de variables.

Cada nodo ejecutará un algoritmo de empaquetado, para optimizar el número de transferencias a realizar en el bus, que transforma los datos antes indicados en objetos de comunicación con un campo de transferencia de datos de 8 bytes tal como indica la Figura 52. El algoritmo que agrupa las variables lo realiza de acuerdo a su tipo, por ejemplo en un objeto pueden ir agrupadas hasta 64 variables tipo bit (si el número de variables es menor el resto del objeto queda libre). Al cambiar de tipo de variable el agrupamiento se inicia siempre en un nuevo objeto.

Las variables que tengan algún tipo de restricción, de ser posible, serán en objetos que tendrán el campo de indicación de restricción habilitado.

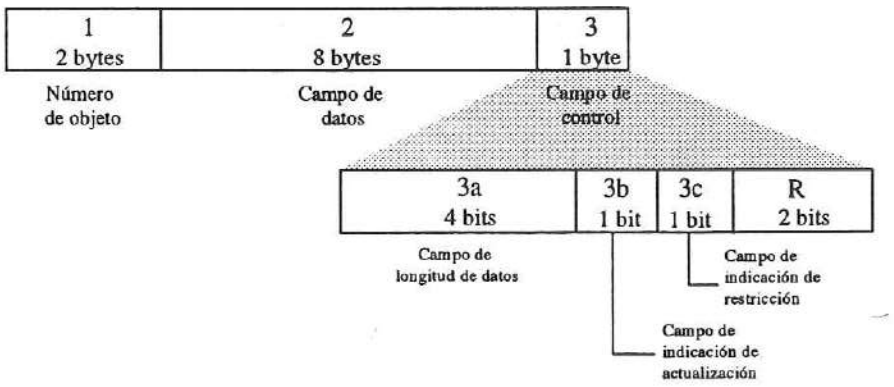


Figura 52: Formato de un objeto de comunicación de un CCC.

La información obtenida de los dispositivos de campo, almacenada en la tabla de objetos de comunicación, será transferida:

- al controlador del bus entre CCCs para su transmisión de acuerdo a las prioridades de los mismos,
- al driver de comunicación con el nivel superior en el CCCP,
- a la consola en el caso de requerir la visualización del estado de las variables locales.

En estos dos últimos casos antes de enviar la información se debe desempaquetar la información de los objetos y relacionarla con cada uno de las variables de la tabla de datos del sistema.

De esta manera la transmisión de las variables integradas será más eficiente, puesto que no se tendrán que transferir una a una sino como datos dentro de un objeto determinado. Las principales características de la tabla de objetos, en donde se puede apreciar su funcionalidad, son las siguientes:

1. Almacenar todos los objetos definidos en el sistema integrado: En la tabla de objetos se mantiene el último valor actualizado de las variables del sistema. La actualización de cada variable la realiza el proceso gestor de *drivers* interactuando con el driver correspondiente del dispositivo integrado. Dependiendo del tipo de la variable, ésta podrá ser modificada por procesos SCADA u otros procesos internos.

2. Proporcionar las variables almacenadas de manera transparente a todos los procesos que las requieran: Debido a que la tabla de objetos es un reflejo de la situación del sistema integrado, otros equipos pueden compartir de manera transparente esta información, sin tener que acceder directamente a los elementos específicos.

Las variables de los dispositivos a integrar - que se encuentren conectados al CCCP - que sólo requieran ser exportadas al sistemas supervisor no originarán transferencias a la red, pues físicamente ya se encontrarán en el CCCP. Las transferencias de las variables (de estos mismos dispositivos) que se requieran intercambiar, con otros nodos, también se encuentran consideradas en los nodos que las envían o reciben respectivamente.

5.9.3 Configuración Remota.

Una vez introducidos todos los datos en cada uno de los CCCs, el CCCP procederá a realizar automáticamente la configuración de todo el sistema. Para ésto interrogará a las estaciones una a una, y éstas le transferirán las tablas de variables locales, y con el mismo algoritmo de empaquetado, realizado por cada uno de los nodos, el CCCP procederá a definir su tabla de objetos de comunicación que intercambiará con cada uno de ellos. Como el CCCP es el que tiene que indicar la configuración final de cada nodo éste debe ser el último en configurarse.

Una vez terminada la interrogación a todos los nodos, el CCC maestro deberá verificar si se cumplirán las restricciones de tiempo impuestas en los nodos. Terminada con éxito esta parte, procede a conectarse nuevamente con cada una de las estaciones para comunicarles los valores que deben asignar a sus objetos-mensajes. Caso contrario, señala e indica cuales son los tiempos de actualización que van a poder ser cumplidos.

Otra posibilidad de configuración podrá ser realizada configurando todo el sistema en el CCCP, luego éste se conectará con cada una de los CCCs y les transferirá la configuración local que deben implementar.

El sistema no entrará en funcionamiento normal hasta que el CCCP verifique que se cumplirán todas las restricciones de tiempo. En el caso que ningún nodo tenga restricciones de tiempo, el CCCP procederá a realizar los cálculos para obtener el tiempo mínimo de actualización de variables. Este dato será mostrado al terminar la configuración.

5.9.4 Funcionamiento normal.

Una vez terminada la configuración remota el sistema iniciará la secuencia de funcionamiento normal, en este caso los nodos podrán ser conectados y desconectados sin ningún tipo de restricción.

Cuando se desconecte un nodo, éste deberá indicar al maestro que saldrá de funcionamiento. El maestro tomará nota de esta situación y no indicará ninguna alarma. En el caso de que el nodo no envíe ningún mensaje, dentro de un tiempo determinado, el maestro indicará una alarma de nodo fuera de servicio.

Cuando se conecta un nodo, éste deberá enviar un mensaje de operación para ser reconocido por el maestro. Este le dará de alta y podrá entonces entrar en funcionamiento normal. Si se han añadido o quitado algunas variables, el esclavo enviará un menaje de operación para que el maestro pueda detectar este cambio y, antes de darle permiso de funcionamiento normal, actualizará los objetos que él guarda de reflejo de este nodo.

5.10 Procesos internos del CCC.

Para proporcionar la funcionalidad requerida en los CCCs estos tendrán definidos una serie de procesos internos (mostrados en la Figura 44) que detallamos a continuación.

5.10.1 Proceso consola.

Este proceso será el primero que arrancará e inicializará a todos los demás según configuración. También realizará la función de comunicación con un equipo de visualización - normalmente local, aunque podría ser remoto - a través de un puerto serie RS-232 para la monitorización de la evolución del sistema. Además, permitirá la entrada de comandos específicos para la transferencia de datos hacia y desde el CCC. Por medio de este canal de comunicación, el operador podrá configurar el sistema para su ampliación o modificación. A través de la consola se podrá acceder al propio sistema operativo.

5.10.2 Procesos gestores.

Asociados a la tabla de objetos, el CCC dispondrá de unos procesos que se encargarán de gestionar:

- las comunicaciones con los *Drivers* de los elementos de campo,
- las alarmas,
- los eventos definidos por el usuario.

Estos procesos son los que se encargarán de mantener siempre actualizada la tabla de objetos. Para agilizar el refresco de la tabla de objetos, los procesos gestores de *drivers* solamente tomarán, del driver asociado, las variables que han sido modificadas. Los gestores de alarmas y eventos comprobarán la situación de algunas variables, definidas según configuración previa, y de acuerdo a su valor originarán una señal de alarma o arrancarán un proceso de emergencia determinado. De esta manera en los CCCs también se podrá definir un control local.

Con la disponibilidad de estos gestores se puede realizar un control redundante de las condiciones de error o fallo, que se pueden presentar en el sistema integrado, según consignas de alarmas establecidas en configuración. Además, con los gestores de eventos, se podrá realizar un seguimiento histórico de determinadas variables e incluso permitir su almacenamiento para su uso posterior por otros procesos, como por ejemplo el de consola.

5.10.3 Procesos de atención al supervisor.

Estos procesos, disponibles sólo en el CCCP, son los que se encargarán de la transferencia a los niveles superiores de la información de todo el sistema integrado. La Figura 53 muestra la forma en que estos procesos realizarán tal transferencia.

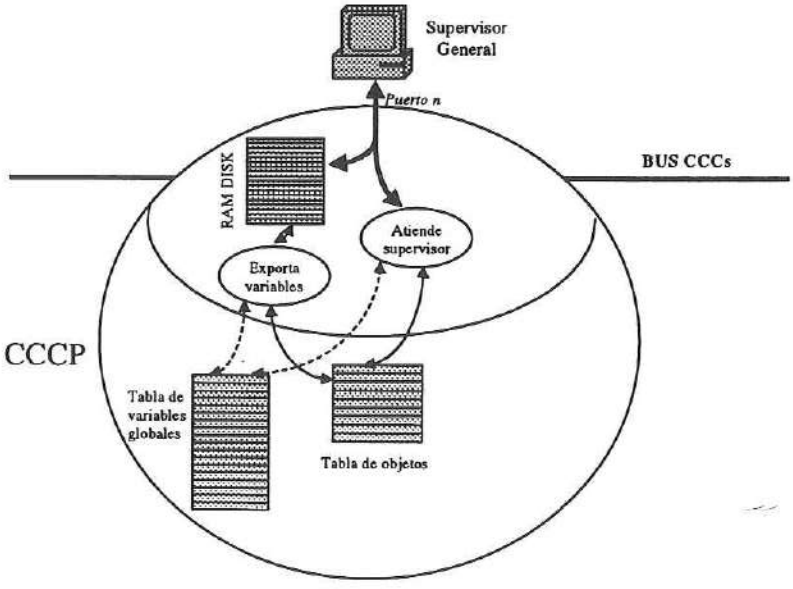


Figura 53: Procesos de atención al supervisor.

La transferencia a los niveles superiores podrá ser re: .ada de alguna de las siguientes formas:

1. Mediante el proceso "exporta variables" que escribirá la información necesaria para el nivel superior en un RAMDISK - haciéndolo exportable de tal manera que la máquina del nivel superior podrá montarlo y acceder a él a través de un protocolo de red como por ejemplo TCP/IP. Para ésto desempaquetará la información de la tabla de objetos, de acuerdo a la tabla de configuración de variables globales (transferidas por cada nodo al momento de la configuración), y actualizará periódicamente la información del RAMDISK.

2. Mediante el proceso "atiende supervisor" que se encargará de gestionar las transferencias con el nivel superior atendiendo el puerto de comunicación por el que se conecta a este nivel con un protocolo específico.

5.10.4 Procesos de transferencia a bus CCCs.

En este apartado se describe cómo se realizará toda la gestión de los envíos y recepciones de los objetos entre CCCs mediante los procesos contenidos en el bloque "procesos de transferencia a bus" de la Figura 44. En estos procesos se le añadirá, o quitará, los campos necesarios para convertir cada elemento de la tabla de objetos en mensaje – objeto CAN, o viceversa, para su transmisión o recepción a través del bus. En la Figura 54 se presentan estos procesos para su implementación en un CCC. También se indica el número de mensaje - objeto de CAN que se utilizará para cada transferencia.

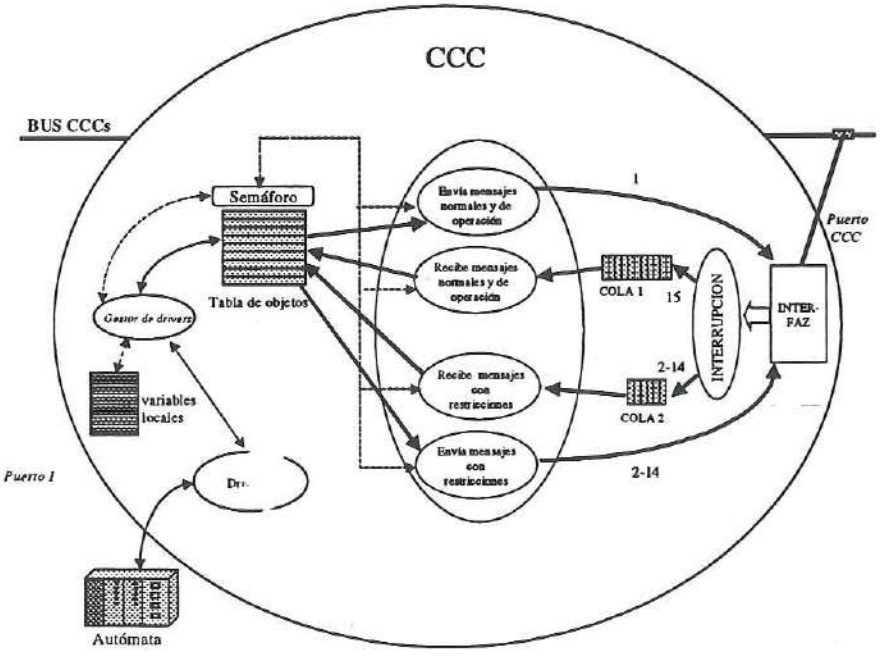


Figura 54: Procesos de transferencia a bus CCCs.

Las transferencias de los objetos, definidos en los CCCs, a la red se realizarán a través de un circuito de interfaz al bus CCC. Para evitar pérdidas en las transferencias todos los objetos - mensajes que lleguen deberán originar una interrupción hardware al CCC. El sistema también deberá implementar dos colas, una para los mensaje-objetos que transferirán datos sin restricciones y otra para los que transferirán datos con restricciones. Para la recepción de la interrupción se deberá implementar un proceso de atención a la misma, que se encargará de escribir el dato que llega en la cola que le corresponda (la selección de la cola se realizará de manera inmediata verificando el número del mensaje-objeto que produjo la interrupción), y enviará una señal de aviso al proceso de recepción oportuno.

Los procesos indicados en la Figura 54 realizan las siguientes funciones:

1. Proceso de envío de objetos de operación y sin restricciones de tiempo.

Este proceso se encargará de tomar los objetos de la tabla que no necesitan cumplir restricciones especiales, y de transmitirlos de acuerdo a la configuración impuesta por el CCCP. La planificación de envío de los objetos que llevan este tipo de variables, podrá ser realizada de dos formas: por interrogación del CCCP, o de manera automática cuando alguna variable que pertenece al objeto cambie. En este trabajo descartamos la interrogación por las razones expuestas en 3.4.2.

2. Proceso de recepción de objetos de operación y sin restricciones de tiempo.

Este proceso, junto con el descrito antes, completará todas las acciones a realizar para mantener un intercambio bidireccional entre los CCCs y el CCCP. Este proceso, cuando el proceso interrupción le avisa, tomará los mensajes - objetos de la cola de recepción y los transferirá a su posición correspondiente dentro de la tabla de objetos del nodo.

3. Proceso de envío de objetos con restricciones.

Los objetos de un CCC que deban cumplir algunas restricciones, por ejemplo envío cíclico, consistencia temporal, etc. serán enviados mediante este proceso.

En el caso de transferencias cíclicas se implementará un algoritmo de planificación de envío, que agrupará a estos objetos de acuerdo a su tiempo cíclico que tengan definido. El sistema sólo aceptará tiempos múltiplos del periodo "t" que dependerá de la velocidad de funcionamiento del bus y de la longitud de este tipo de objetos. En la Figura 55 se indica un ejemplo de la forma en que el algoritmo agrupará los objetos (en este ejemplo se han considerado 3 variables cíclicas con tiempos de transmisión diversos – A con periodo "t", B con periodo "2t" y C con periodo "3t" -). En este ejemplo el programa asignará 6 grupos dentro del macrociclo (el macrociclo indica el tiempo necesario para repetir una secuencia completa de envío).

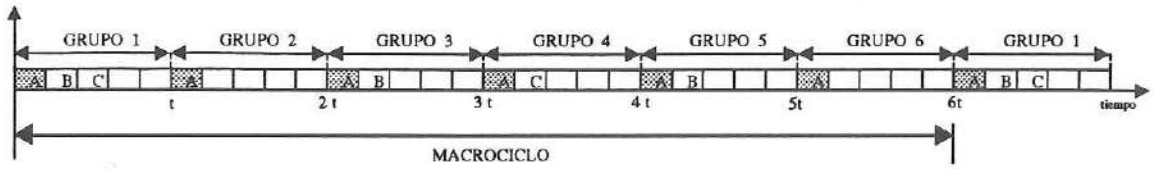


Figura 55: Ejemplo de secuencia de envío de variables con restricción de tiempo.

Inicialmente, en el momento de la configuración, este proceso ordenará los objetos con restricciones calculando el número de grupos y los ciclos de tiempo (t) que son necesarios para completar un macrociclo, luego procederá a leer, actuando sobre el semáforo, todas las variables del grupo que corresponde enviar y colocará los valores dentro de los objetos correspondientes (predefinidos por el CCCP) del circuito de interfaz.

Para comprobar que todas las variables ya han sido transmitidas, realizará un OR de todos los bits de indicación de transmisión de cada uno de los objetos asociados al grupo (la indicación de transmisión de todas las variables se obtiene cuando este OR dé como resultado cero). Una vez transmitidos todos los objetos, se comprobará si ya pasó el tiempo para el siguiente grupo. Igualmente se procederá, mientras el sistema esté activo, con todos los grupos de manera indefinida.

4. Proceso recibe mensajes con restricciones.

Este proceso, junto con el descrito antes, completará todas las acciones a realizar para los intercambios entre los CCCs de mensajes con algún tipo de restricción. Este proceso, cuando el proceso interrupción le avisa, tomará los mensajes - objetos de la cola de recepción correspondiente y los transferirá a su posición dentro de la tabla de objetos del nodo.

5.10.5 *Drivers* de comunicaciones.

La función que realizarán los *Drivers* de Comunicaciones será la de establecer las comunicaciones con los elementos de campo, implementando los protocolos necesarios a través de los medios físicos de conexión adecuados.

La complejidad del *Driver* dependerá fundamentalmente del equipo al que esté conectado. Gracias a la actual tendencia de los fabricantes, de ofrecer sistemas abiertos, es posible disponer de ellos sin mayor problema a bajo coste. Debido a que los *Drivers* son implementados como procesos independientes, la funcionalidad del sistema no se ve alterada por el número y tipo de comunicaciones realizadas, solamente se ralentizan un tanto las tareas de la CPU.

En el caso de drivers comerciales éstos serán atendidos por el proceso "gestor de drivers" para adecuar la forma de presentar los datos a los requerimientos del sistema. Si los drivers son propios del CCC éstos escribirán directamente en la tabla de objetos.

5.11 Sistema de comunicación entre los procesos del CCC.

Todos los procesos que se definan en un CCC, más las funciones de control internas necesitan compartir los datos de los elementos de campo que el CCC integra. Para lograr ésto se empleará alguno de los mecanismo de comunicación entre procesos, por ejemplo memoria compartida, señales, etc., que muchos sistemas en tiempo real, como VRTX/OS, VxWorks, pSOS+, QNX, OS9, OS9000, entre otros ofrecen.

Uno de los mecanismos más empleados es el de generar un área de memoria a la que podrán acceder todos los procesos, definidos anteriormente, que lo requieran. Como esta área es solicitada por varios procesos debe existir un mecanismo para realizar la sincronización entre los mismos. En este caso el sincronismo se realiza por medio de un semáforo. El esquema de funcionamiento simplificado de la zona de memoria compartida se indica en la Figura 56.

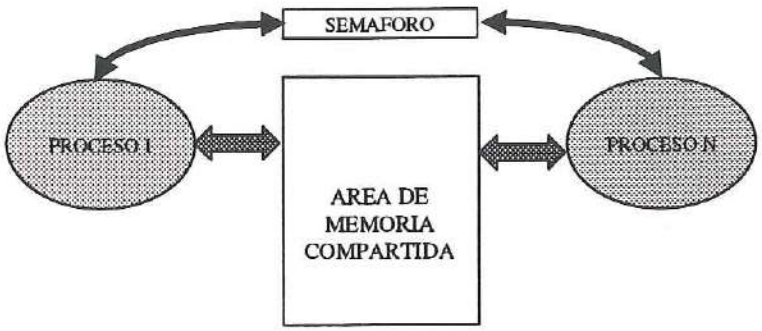


Figura 56: Esquema simplificado de funcionamiento del área de memoria compartida.

5.12 Cálculo de tiempos de retardo de los mensajes en la red de interconexión de CCCs sobre CAN.

5.12.1 Parámetros característicos.

Como ya hemos definido en el apartado 5.7, para comunicar los CCCs haremos uso de un protocolo que tiene como base, en los niveles 1 y 2 de OSI, el estándar CAN [CAN2.0].

Uno de los parámetros que caracteriza al bus entre CCCs, al igual que a todos los buses tipo serie, es el tiempo de retardo - tiempo transcurrido desde que un dato cambia en un nodo del bus hasta que es registrado en otro nodo del mismo -. Este tiempo, que puede ser expresado por la Ecuación 8, depende de la longitud de los datos del protocolo (longitud de trama), la velocidad de transmisión y de un tiempo adicional - el tiempo de *overhead* -, que contiene el retardo de tiempo del software, el retardo de tiempo del controlador y el tiempo de acceso al bus.

$$T_r = \frac{LT}{V} + OT \qquad \text{Ecuación 8}$$

Donde:

T_r = Tiempo de retardo
LT = Longitud de trama en bits
V = Velocidad en b/s
OT = Tiempo adicional de *overhead*

La longitud de la trama es el factor más importante que influye en el retardo de tiempo La trama de datos contiene información, añadida a los datos del usuario, para sincronizar, para identificar, para controlar y para garantizar el flujo de datos. Por lo tanto no sólo el formato de los datos del usuario sino también esta información determina la longitud de la trama.

La velocidad es otro parámetro importante para el valor del retardo de tiempo, y depende significativamente de la distancia de transmisión. En los puntos siguientes se presenta un análisis para calcular la parte más importante del tiempo de retardo, es decir el tiempo para transmitir una trama de datos por el bus entre CCCs.

Es de resaltar que para el análisis se han considerado las situaciones más usuales dentro de las aplicaciones industriales, esto es velocidad máxima en función de la distancia de acuerdo con el estándar ISO 11898/2/ y el CiA DS 102-1/2/.

5.12.2 Aspectos teóricos del tiempo de retardo.

Con el valor de los bits de relleno, dado por la Ecuación 3, podemos calcular que el tiempo para trasmitir un mensaje m, con campo de arbitraje estándar, estará limitado por la siguiente expresión:

$$\frac{44+8\times L_D}{d_a} \le T_m \le \frac{44+8\times L_D+\left\lfloor\frac{34+8\times L_D-1}{4}\right\rfloor}{d_a} \qquad \text{Ecuación 9}$$

En donde:

L_D : Indica la longitud de los datos (de 0 a 8 bytes).
d_a : Indica la velocidad de transmisión en b/s.
T_m : Indica el tiempo empleado para transmitir el mensaje m.

De la expresión anterior podemos tomar el límite máximo, que nos indicara el caso más desfavorable, de tiempo de transmisión del mensaje m.

$$C_m = \left(44 + 3 + 8 \times L_D + \left\lceil \frac{33 + 8 \times L_D}{4} \right\rceil \right) \times \tau_{bit} \qquad \text{Ecuación 10}$$

En donde:

C_m : Tiempo máximo (más desfavorable), de transferencia de la trama del mensaje (m).

τ_{bit} : Indica el tiempo empleado para transmitir un bit $\frac{1}{d_u}$

Por ejemplo si consideramos una velocidad de 100 kb/s y 1 Mb/s y un bloque de datos de 8 bytes encontramos que el tiempo máximo empleado para transmitir este mensaje es:

Velocidad de transmisión	Identificador Estándar	Identificador Extendido
100 kb/s ($\tau_{bit} = 10\mu s$)	1.35 ms	1.6 ms
1 Mb/s ($\tau_{bit} = 1\mu s$)	135 µs	160 µs

Tabla 39: Tiempo empleado para transmitir un mensaje entre CCCs

La trama remota de petición no contiene datos; éstos llegarán en el mensaje de respuesta. Así el tiempo empleado para obtener los datos, por petición remota en mensajes con identificador estándar, puede calcularse con la Ecuación 11. Se ha supuesto que se envía el mensaje de respuesta inmediatamente después de que termina la petición.

$$\frac{47 + (44 + 8 \times L_D)}{d_u} \leq T_{mR} \leq \frac{47 + 0{,}25 \times (34 - 1) + (44 + 8 \times L_D + 0{,}25 \times (34 + 8 \times L_D - 1))}{d_u} \qquad \text{Ecuación 11}$$

En donde:

T_{mR} : Indica el tiempo empleado para transmitir el mensaje de petición remota mR.

De esta ecuación, al igual que antes, tomamos la expresión que da el tiempo máximo, empleado por cada uno de los mensajes de petición remota, como valor del caso más desfavorable de tiempo de transferencia del mensaje (i) sobre la red CAN:

$$C_{mR} = \left(144 + 8 \times L_D + \left\lceil \frac{53 + 8 \times L_D}{4} \right\rceil \right) \times \tau_{bit} \qquad \text{Ecuación 12}$$

En donde:

C_{mR} : Tiempo máximo, valor más desfavorable, de transferencia del mensaje remoto (mR).

5.12.3 Cálculo del tiempo de retardo en la red de CCCs.

Los estudios realizados para el cálculo del tiempo de retardo en un bus CAN [TinBur94] aplican el análisis desarrollado para planificación de procesadores en tiempo real con prioridad fija [AuBu93] [BuNi93].

Antes de presentar el análisis adaptado al uso propuesto de CAN con la red de CCCs definimos algunos términos. Un **mensaje** es un mensaje CAN asignado a un identificador único y contiene de 0 a 8 bytes de datos. Se asume que un mensaje dado es puesto en la cola de transmisión de manera cíclica, es decir el mensaje es colocado a intervalos, con el mismo tamaño y con el mismo identificador. Un mensaje dado es puesto en la cola de una estación dentro de una **ventana de puesta en cola**, con un intervalo mínimo entre las subsecuentes ventanas en la cola (los mensajes no tienen que ser estrictamente periódicos: un mensaje puede ser esporádico, pero debe haber un tiempo mínimo entre la puesta en cola del mensaje). Esto se ilustra en la Figura 57.

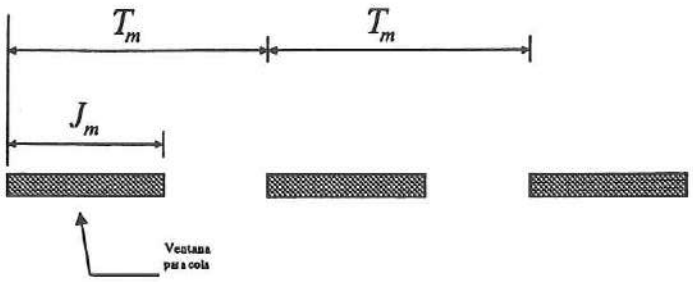


Figura 57: Puesta en cola de un mensaje periódico.

El **periodo de un mensaje** dado m es denotado como T_m. El ancho de la ventana de puesta en cola para el mensaje m (es decir el *jitter* en la puesta en cola del mensaje) está expresado por J_m. El término b_m define el número de bytes en el mensaje; C_m denota el peor tiempo tomado para transmitir físicamente el mensaje sobre el bus indicado por la Ecuación 10. Este no incluye los retardos debido a la contención del bus; incluye sólo el tiempo tomado para transmitir el campo de identificación, otros campos del mensaje (tales como CRC), y los datos del mismo mensaje, por lo tanto C_m es función de b_m.

Dado que CAN es primordialmente un bus basado en prioridades, se le puede aplicar directamente la teoría de sistemas donde las actividades son despachadas de acuerdo a prioridades fijas [Tin93] [Tin94], obteniendo la expresión de la Ecuación 13.

$$R_m = J_m + w_m + C_m$$

Ecuación 13

En donde R_m indica el tiempo más largo entre el tiempo de puesta en cola del mensaje m y el tiempo de llegada a la estación de destino. Para que el mensaje m sea planificable este tiempo tiene que ser menor, o al menos igual, que el tiempo máximo D_m que tiene asignado para su transmisión.

El término w_m es el tiempo más largo de retardo que puede permanecer en la cola el mensaje m (debido a los retardos que le puedan originar los mensajes de más baja prioridad – sólo cuando alguno de éstos ya se encuentre ocupando el bus -, y los de más alta prioridad que siempre le harán esperar).

Aplicando el método propuesto en [AuBu93] encontramos que el retardo que experimenta en la cola el mensaje m está dado por la Ecuación 14.

$$w_m = B_m + \sum_{\forall j \in hp(m)} \left\lceil \frac{w_m + J_j + \tau_{bit}}{T_j} \right\rceil C_j \qquad \text{Ecuación 14}$$

El conjunto *hp(m)* agrupa a todos los mensajes de mayor prioridad que el mensaje m. T_j es el periodo de un mensaje (j) dado, J_j es el *jitter* de puesta en cola del mensaje, y B_m es el tiempo más largo que el mensaje m puede ser retrasado por mensajes de menor prioridad que él (este tiempo es igual al tiempo empleado para transmitir el mensaje de menor prioridad más largo), y puede ser definido por la Ecuación 15.

$$B_m = \underset{\forall k \in lp(m)}{Máx} (C_k) \qquad \text{Ecuación 15}$$

Donde *lp(m)* es el conjunto de variables menos prioritarias que m. Hay que tener en cuenta que si existe un número no determinado de variables, de las que no podemos definir su tamaño, debemos hacer que este valor sea el máximo (de la Tabla 39 deducimos que es igual a $135\tau_{bit}$). Para encontrar la solución de la ecuación anterior se puede utilizar la relación recurrente siguiente:

$$w_m^{n+1} = B_m + \sum_{\forall j \in hp(m)} \left\lceil \frac{w_m^n + J_j + \tau_{bit}}{T_j} \right\rceil C_j \qquad \text{Ecuación 16}$$

Se puede utilizar el valor inicial $w_m^0 = 0$, y proceder al cálculo hasta que la solución converja, es decir cuando $w_m^{n+1} = w_m^n$.

La Ecuación 16 no considera cómo se seleccionan los identificadores (y por lo tanto la prioridad). Sin embargo de los trabajos propuestos en [AuBu93] [LeWh82] se puede conocer que el ordenamiento óptimo se obtiene cuando se les asigna las prioridades mayores a las tareas que tienen el menor valor de $(D_m - J_m)$.

Con la Ecuación 16 podemos calcular cuál es el máximo tiempo de actualización del último objeto enviado por el nodo que tiene el mensaje menos prioritario, al nodo CCC principal – determinando de esta manera el tiempo máximo de actualización de todas las variables del sistema integrado con esta metodología -; o bien verificar si las restricciones de tiempo impuestas por los nodos se cumplen.

5.12.4 Programa para el cálculo de los tiempos máximos de retardo y verificación del cumplimiento de las restricciones temporales (PCTMR).

Como herramienta de ayuda se ha desarrollado un programa que permite calcular, según el análisis de 5.12.4, los tiempos máximos de retardo que presentarán las transferencias entre CCCs. El programa también es ejecutado por el CCCP para verificar si todos los nodos cumplen las restricciones temporales impuestas.

Los mensajes del sistema, con todas sus características, son leídos directamente de un fichero de entrada como el mostrado en la Figura 58. El programa para el cálculo de los tiempos máximos de retardo (**PCTMR**), pide en el momento de inicializarse la velocidad a que operará el bus.

En la Figura 59 se muestra un ejemplo de fichero de salida del PCTMR. Como resultado muestra el tiempo máximo empleado por cada uno de los mensajes. Por ejemplo el mensaje más prioritario, el cero, que se planifica para envío cada 10 ms, tiene un tiempo de retardo de sólo 1.683 ms. Para los mensajes a partir del 316, ya no se cumplen las restricciones de tiempo impuestas. Por ejemplo el mensaje 317, al cual se le había asignado una periodicidad de 400 ms, no se puede enviar antes de 478,723 ms. En el tiempo que él tenía signado es imposible enviarlo.

```
/* Fichero de ingreso datos a PCTMR */

numero de mensajes = 593

Prioridad   Nombre var.          T          n.bytes      D       J
-------------------------------------------------------------------
    0       Bomba_123          10000           2        5000     3
    1       Agita_44           10000           2       10000     3
    2       Valvula_24         10000           2        8000     3
    3       Tempe_56           20000           2       10000     3
    4       Motor_23           20000           2       15000     3
    5       Solen_564          20000           2       18000     3
.
.
.
  589       Indicad_1       40000000           8      401003     3
  590       Nivel_33        40000000           8      401003     3
  591       Bomba_67        40000000           8      401003     3
  592       PH_85           40000000           8      401003     3
```

Figura 58: Ejemplo de fichero de entrada de datos.

Del último ejemplo podemos deducir que el programa también puede servir para encontrar los tiempos en los cuales todos los mensajes son planificables. De hecho este método es el que utiliza el CCCP para encontrar los tiempos en que se pueden enviar todos los mensajes.

```
--- RESULTADOS en usg. ---

                    Velocidad del bus:    125.000 Kbit/s
          Longitud máxima de los mensajes:  8 bytes
                    Número de mensajes:     593

Prior Identificador   T        n.bytes  C      D       J   R          Comentario
-----------------------------------------------------------------------------------
  0   Bomba_123       10000    2        128     5000   3   1683
  1   Agita_44        10000    2        128    10000   3   2283
  2   Valvula_24      10000    2        128     8000   3   2883
  3   Tempe_56        20000    2        128    10000   3   3483
  4   Motor_23        20000    2        128    15000   3   4083
  5   Solen_564       20000    2        128    18000   3   4683

313   Presion_1       40000000 8        512   401003   3   398123
314   Presion_2       40000000 8        512   401003   3   399203
315   Presion_3       40000000 8        512   401003   3   400283
316   Presion_4       40000000 8        512   401003   3   ( 477643 ) -> R > D
317   Presion_5       40000000 8        512   401003   3   ( 478723 ) -> R > D


589   Indicad_1       40000000 8        512   401003   3   ( 956763 ) -> R > D
590   Nivel_33        40000000 8        512   401003   3   ( 957843 ) -> R > D
591   Bomba_67        40000000 8        512   401003   3   ( 958923 ) -> R > D
592   PH_85           40000000 8        512   401003   3   ( 960003 ) -> R > D
-----------------------------------------------------------------------------------
```

Figura 59: Ejemplo de fichero obtenido con el PCTMR.

5.12.5 Estimación de los tiempos de retardo del software y del controlador.

Para completar el análisis de tiempos del sistema, hace falta considerar los tiempos empleados para ejecutar el software de control en cada CCC, y los tiempos de retardo que presente el controlador CAN utilizado.

El cálculo del tiempo de latencia de una aplicación software [Henn93], en un sistema operativo multitarea en tiempo real, es un tema de investigación abierto. No obstante se puede realizar una estimación cuantitativa del tiempo de retardo del software, teniendo en cuenta la frecuencia del reloj interno del ordenador (ciclos de reloj), de la siguiente manera:

$$T_{CPU} = CRP \times \tau_{CR} \qquad \text{Ecuación 17}$$

Donde:

T_{CPU} : Tiempo de CPU.
CRP: Ciclos de Reloj que emplea el programa determinado.
τ_{CR} : Duración de un ciclo de reloj.

También puede expresarse en función de las instrucciones ejecutadas por la CPU como indica la Ecuación 18.

$$T_{CPU} = \sum_{i=1}^{n} (CPI_i \times RI_i) \times \tau_{CR} \qquad \text{Ecuación 18}$$

Donde:

CPI_i : Indica el número medio de ciclos de reloj para la instrucción i.

RI_i : Indica el número de veces que se ejecuta la instrucción i dentro del programa.

n: Indica el número total de instrucciones diferentes utilizadas en el programa.

Este tiempo está sujeto a las características del hardware o a la habilidad del programador, y depende de parámetros como el tipo de compilador utilizado, etc., por lo tanto solamente nos limitaremos a realizar una estimación del mismo.

Para realizar la estimación del tiempo de respuesta del software y controlador (TRS), hacemos uso de la funcionalidad de los objetos mensajes de CAN, de poder realizar una petición remota de un objeto definido en otro nodo. El esquema empleado para realizar la prueba se muestra en la Figura 60.

El procedimiento consiste en hacer una petición remota, desde el CCC de medida, que origine una interrupción al programa de la CPU bajo prueba. Éste procederá a atender la interrupción con el proceso de atención respectivo, incluido en el programa bajo prueba que tendrá instalado. El programa que ejecuta tiene la funcionalidad descrita en 5.6, y se le realizan varias pruebas, variando el número de drivers y manteniendo constantes el número de procesos gestores. Una vez que el driver correspondiente tenga el valor de la variable pedida, se coloca ésta en el objeto de respuesta correspondiente y se envía al nodo que la solicitó.

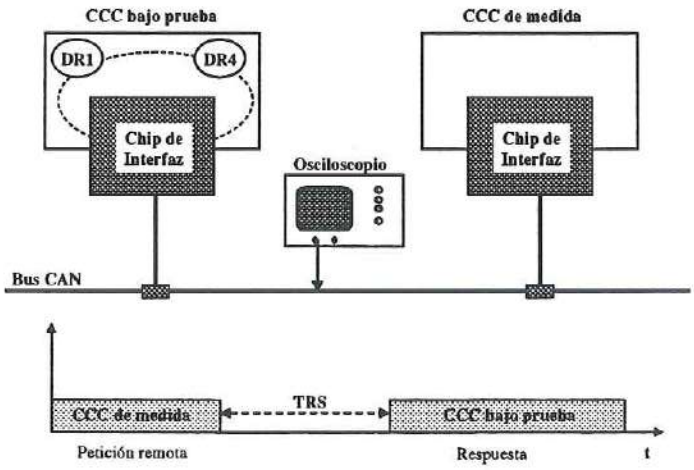


Figura 60: Esquema utilizado en la medición del TRS.

De esta manera tendremos una estimación aceptable del tiempo de retardo del software, más el empleado por el controlador CAN.

5.13 Resultados obtenidos.

Presentamos a continuación los resultados teóricos de la implantación metodología propuesta en cuanto a tiempos y flexibilidad de planificación. Se han considerado sólo algunos casos típicos que encontramos al implantarla, ya que la configuración del sistema depende de la cantidad y distribución física de los elementos que se integran, y por lo tanto existen muchos casos posibles.

Utilizando el programa propuesto en 5.12.4 (PCTMR) se han simulado las condiciones más extremas en las que podrá encontrarse una red de CCCs. El tiempo de retardo del software y del controlador, no influirán en los resultados siempre y cuando éste sea menor que el tiempo que tendrá el CCC entre trama y trama (en el peor de los casos llegada de datos en ráfaga). Este tiempo dependerá de la velocidad del bus y del tiempo que tarde el controlador de bus en indicar que tiene un nuevo dato. La Figura 61 indica el tiempo que transcurre, desde que llega el primer bit de la trama hasta que el controlador lo interrumpe (con el último bit del campo ACK de la trama). Como en el sistema propuesto la trama más corta tendrá 1 byte de datos (n=1) el tiempo total disponible estará indicado por la Ecuación 19.

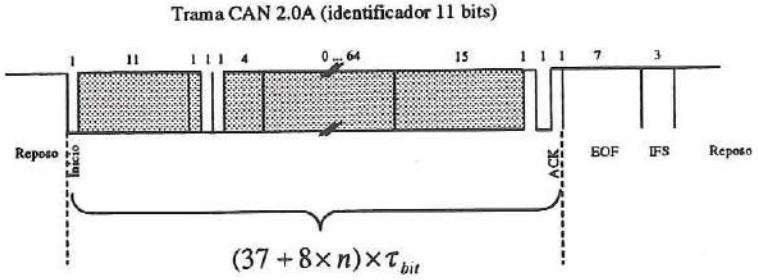


Figura 61: Bits de trama antes de interrupción.

$$45\times\tau_{bit}+10\times\tau_{Bbit} \qquad \text{Ecuación 19}$$

El segundo término se debe a los 10 bits de los campos EOF (7 bits) e IFS (3 bits). Con la Ecuación 19 se encuentran los tiempos libres de los CCCs, entre trama y trama, mostrados en la Tabla 40.

Velocidad	τ_{bit}	Tiempo libre
1 Mb/s	1 µs	55 µs
500 kb/s	2 µs	110 µs
250 kb/s	4 µs	220 µs
125 kb/s	8 µs	440 µs

Tabla 40: Tiempos libres para los CCCs entre tramas.

Los resultados que se mostrarán en los apartados siguientes han sido comprobados luego con el sistema de demostración montado en el Laboratorio de Equipos y Productos Industriales del Centro de Microelectrónica Aplicada (CMA) de la ULPGC.

5.13.1 Tiempos máximos de actualización de la Tabla General de Objetos en una red de CCCs cuando no se incluyen mensajes con restricciones.

Para esta simulación suponemos que los nodos trabajan en condiciones extremas, es decir con todas las tablas completas al máximo, pero sin mensajes entre nodos que deban cumplir algún tipo de restricción. Este es el caso de integración de un sistema en donde sólo interesa enviar y recibir los valores de las variables de los dispositivos integrados. El tiempo que se calcula es el tiempo máximo que podría tardar el último objeto del nodo con el más alto identificador - nodo de menor prioridad -.

Bajo esta suposición el número total de mensajes se muestra en la Tabla 41, en donde "n" es el número de nodos sin considerar el CCCP.

Campo de control			Campo de identificación y operación								Número de mensajes
ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0	Bits del identificador
0	0	0									Reservado
0	0	1	Número del mensaje 0 – 255								0
0	1	0	Número del nodo 0 - 15				Tipo de operación				0
0	1	1	Número del nodo 0 - 15				Número de trozo 0 – 15				$16\times n$
1	0	0	Número del nodo 0 - 15				Tipo de operación				0
1	0	1	Número del nodo 0 - 15				Número de trozo 0 – 15				$16\times n$
1	1	0									Reservado
1	1	1	No permitido por el estándar								
Número total de mensajes en la red											$32\times n$

Tabla 41: Número total de mensajes.

Los resultados que muestran el peor tiempo de actualización de todos los datos, obtenidos con valores distintos de velocidad y número de nodos (el número de bytes de datos se fijó en 8), se muestran en la Tabla 42. Estos resultados se representan también gráficamente en la Figura 62.

Número total de nodos	2	4	8	16
Número total de mensajes	32	96	224	480
20 kb/s	222,753	654,753	1518,753	3246,753
125 kb/s	35,643	104,763	243,003	519,483
0.5 Mb/s	8,913	26,193	60,753	129,873
1 Mb/s	4,458	13,098	30,378	64,938

Tabla 42: Tiempos máximos de respuesta en ms.

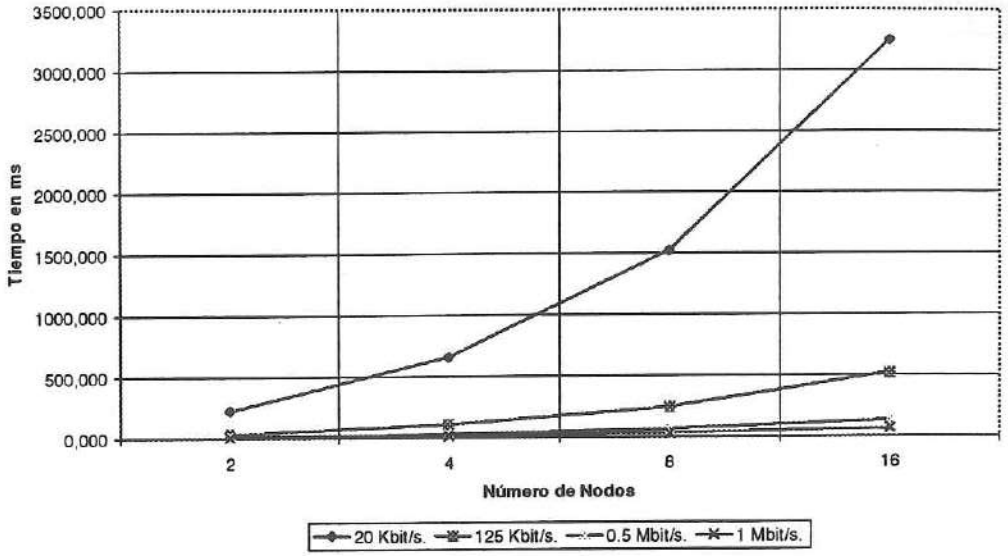


Figura 62: Tiempos máximos de respuesta.

5.13.2 Variación del tiempo máximo de actualización de la tabla general de objetos, cuando se incluyen mensajes con restricciones entre nodos.

Se presentan los resultados obtenidos cuando a un determinado número de nodos se les agrega, además de los datos de transferencia a CCCP, el número máximo de mensajes con restricciones, más un mensaje de difusión del CCCP para indicar los nodos activos y sincronizar los relojes. El número máximo de mensajes con consistencia temporal disponibles en cada nodo es 12. Por lo tanto el número máximo de variables con consistencia que pueden existir en la red, será igual a 6 por el número de nodos totales incluyendo al CCCP (se considera 6, pues en el más desfavorable de los casos, sólo se podrán definir la mitad de recepción y la mitad de transmisión). En la Tabla 43 se indican los mensajes antes mencionados.

Campo de control			Campo de identificación y operación								Número de mensajes
ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0	Bits del identificador
0	0	0									Reservado
0	0	1	Número del mensaje 0 – 255								$1+6\times(n+1)$
0	1	0	Número del nodo 0 – 15				Tipo de operación				0
0	1	1	Número del nodo 0 – 15				Número de trozo 0 - 15				$16\times n$
1	0	0	Número del nodo 0 – 15				Tipo de operación				0
1	0	1	Número del nodo 0 – 15				Número de trozo 0 - 15				$16\times n$
1	1	0									Reservado
1	1	1	No permitido por el estándar								
Número total de mensajes en la red											$7+(38\times n)$

Tabla 43: Número total de mensajes.

Los resultados que muestran el peor tiempo de actualización de todos los datos se muestran en la Tabla 44. Fueron obtenidos con valores distintos de velocidad y número de nodos. El número de bytes de datos se fijó en 8, el de las variables con restricciones en 2. Los resultados se representan gráficamente en la Figura 63.

Número total de nodos	2	4	8	16
Número total de mensajes	45	121	273	577
20 kb/s	271,503	748,503	1702,503	3610,503
125 kb/s	43,443	119,763	272,403	577,683
0.5 Mb/s	10,863	29,943	68,102	144,423
1 Mb/s	5,433	14,973	34,053	72,213

Tabla 44: Tiempos máximos de respuesta en ms.

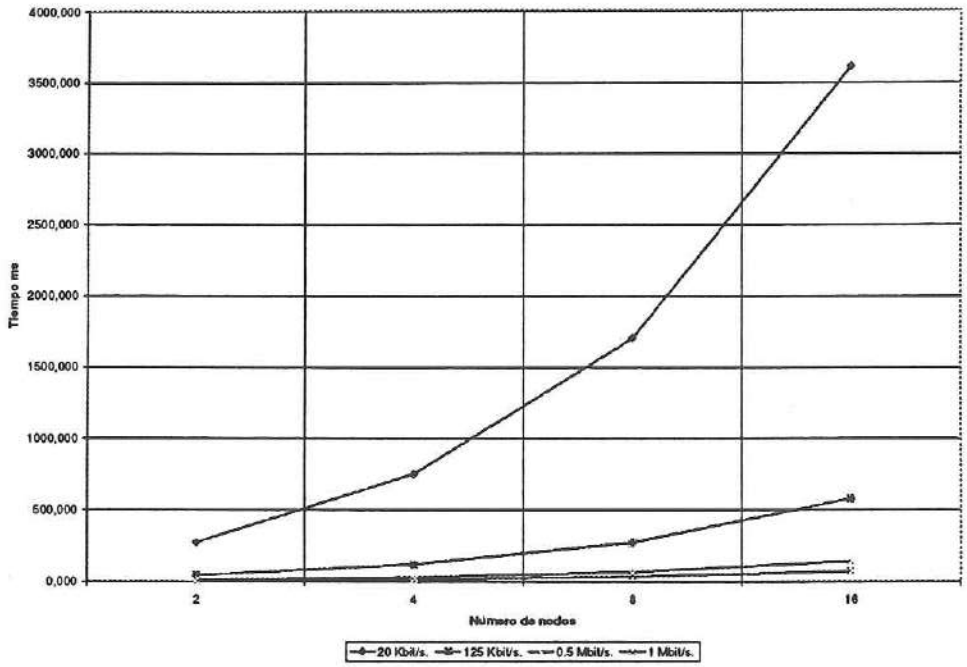


Figura 63: Tiempos máximos de respuesta.

5.13.3 Tiempos de actualización para 16 nodos con número de mensajes de tabla máximos y mensajes prioritarios variables.

Los resultados se han obtenido para el caso de 16 nodos, con transferencia máxima de tablas, para distintos números de mensajes con restricciones. Se presentan dos casos, el primero con longitudes de los datos, de los mensajes con restricciones, de 2 bytes, y el segundo con longitudes de 1 byte.

Tabla	480 mensajes normales de tabla (8 Bytes).											
Mensajes con restricciones	1 mensaje de difusión del maestro (2 Bytes)											
	16		32		48		64		80		96	
	1 Byte	2 Bytes	1 Byte	2 Bytes	1 Byte	2 Bytes	1 Byte	2 Bytes	1 Byte	2 Bytes	1 Byte	2 Bytes
20 k	3302,503	3310,503	3354,503	3370,503	3406,503	3430,503	3458,503	3490,503	3510,503	3550,503	3562,503	3610,503
125 k	528,403	529,683	536,723	539,283	545,043	548,883	553,363	558,483	561,683	568,083	570,003	577,683
0,5 M	132,103	132,423	134,183	134,823	136,263	137,223	138,343	139,623	140,423	142,023	142,503	144,423
1 M	66,053	66,213	67,093	67,413	68,133	68,613	69,173	69,813	70,213	71,013	71,253	72,213

Tabla 45: Tiempos máximos de actualización.

Las gráficas de la Figura 64 muestran el caso de variables con restricciones con longitud de datos de 1 byte. En la Figura 65 se comparan los casos de 1 y 2 bytes; para tener mayor detalle en el gráfico sólo se representan las velocidades de 125 kb/s, 500 kb/s. y 1 Mb/s.

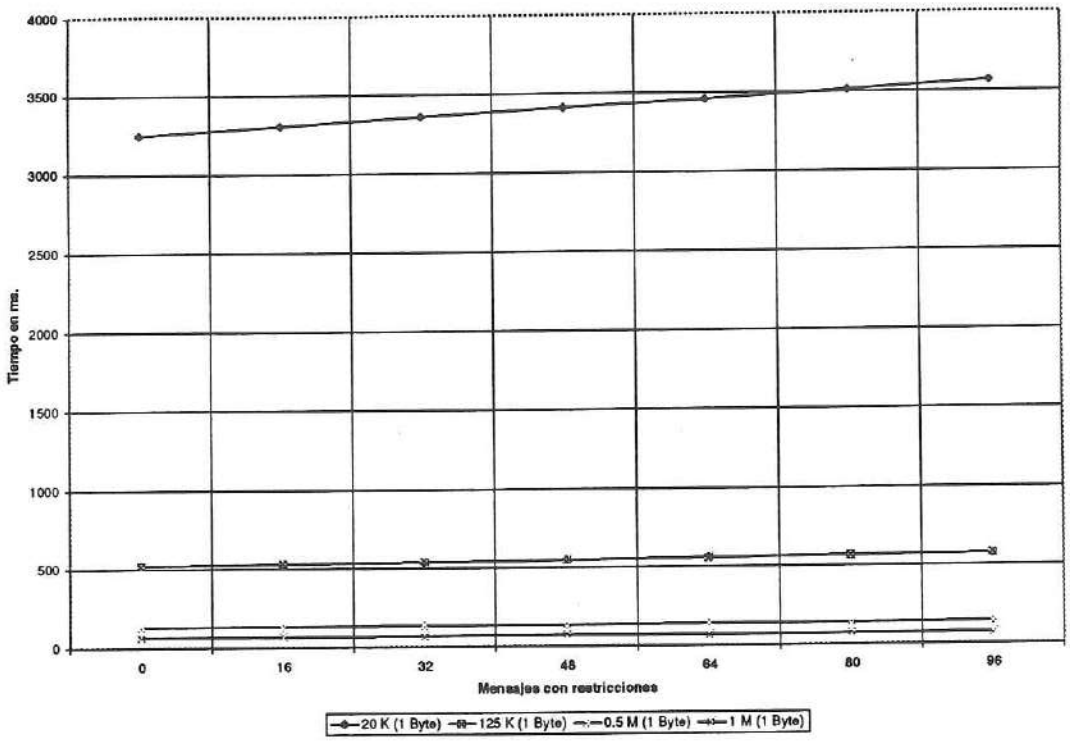


Figura 64: Tiempos máximos de respuesta.

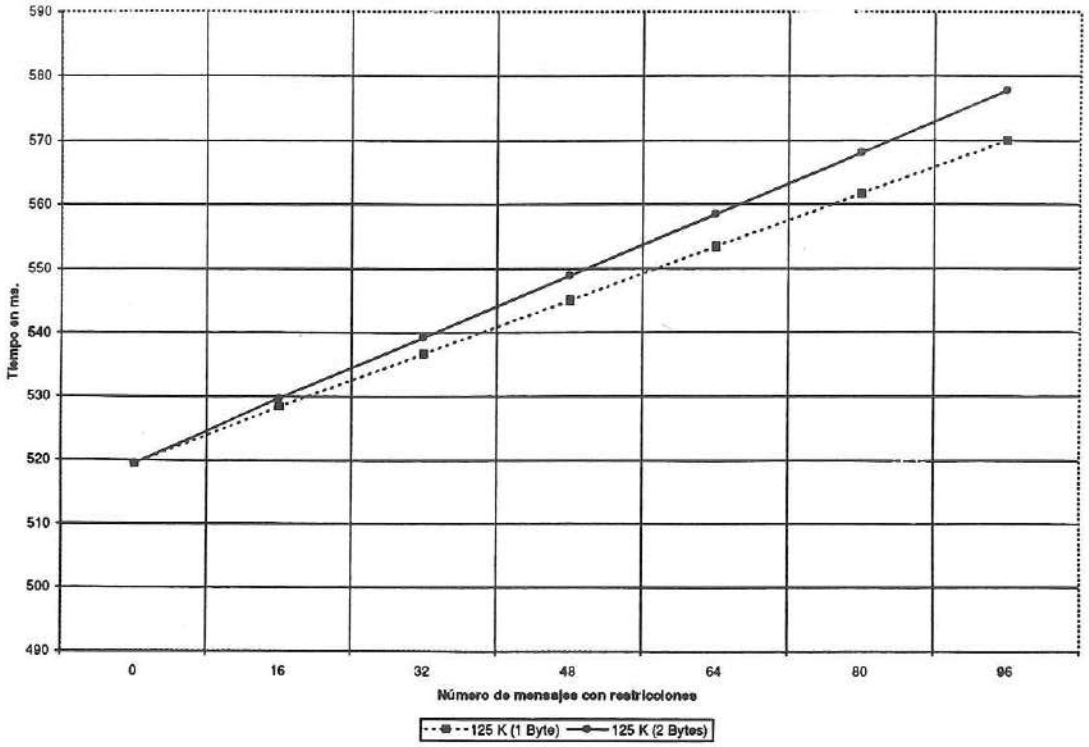


Figura 65: Comparación de tiempos para longitudes de los datos de los mensajes con restricciones de 1 y 2 bytes respectivamente (a 125 kb/s).

5.13.4 Casos con variables cíclicas.

El número de posibilidades de transferencias cíclicas, entre un número determinado de nodos, es ilimitada. A continuación presentamos dos ejemplos para ilustrar que este tipo de mensajes también pueden ser garantizados la red de CCCs propuesta en la metodología.

5.13.4.1 Transferencia de tabla completa y una variable cíclica.

Se presentan los resultados par distintos números de nodos, con transferencia de tabla al máximo, incluyendo una variable cíclica cada 5 ms, tal como indica la Figura 66.

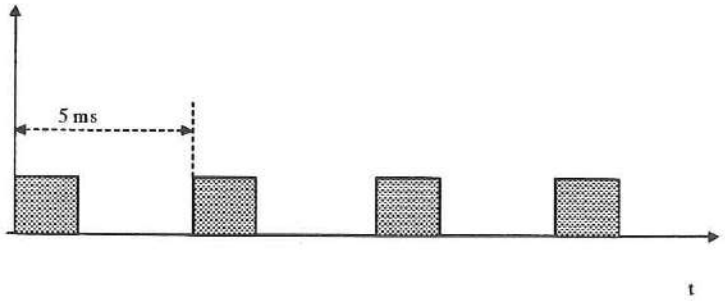


Figura 66: Restricción cíclica de tiempo para una variable

Los resultados se muestran en la Tabla 46 y la Figura 67.

Nodos	2	4	8	16
125 kb/s	40,443	119,163	276,003	590,282
0.5 Mb/s	9,213	27,093	62,703	133,92
1 Mb/s	4,533	13,323	30,9	65,988

Tabla 46: Tiempos de respuesta (ms) para tabla más un mensaje con restricción.

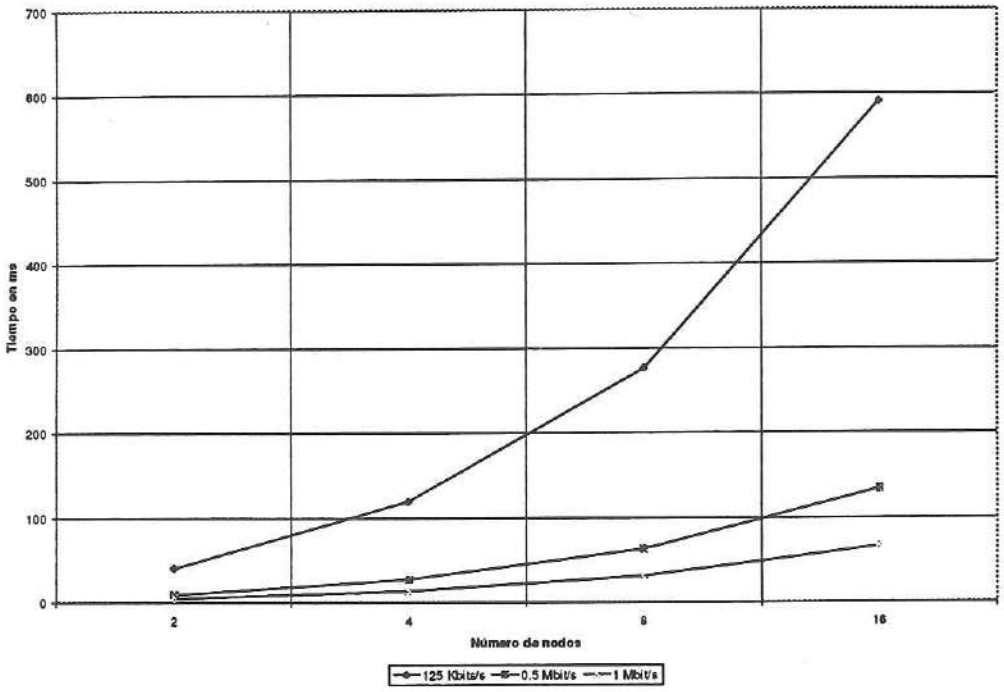


Figura 67: Tiempo máximo de respuesta con una variable con restricción.

5.13.4.2 Transferencia de tabla completa y dos variables cíclicas.

Este caso es similar al anterior pero con dos variables cíclicas con diferentes periodos, una cada 5 ms y otra cada 10 ms. La Figura 68 indica este caso; y la Tabla 47 y la Figura 69 muestran los resultados.

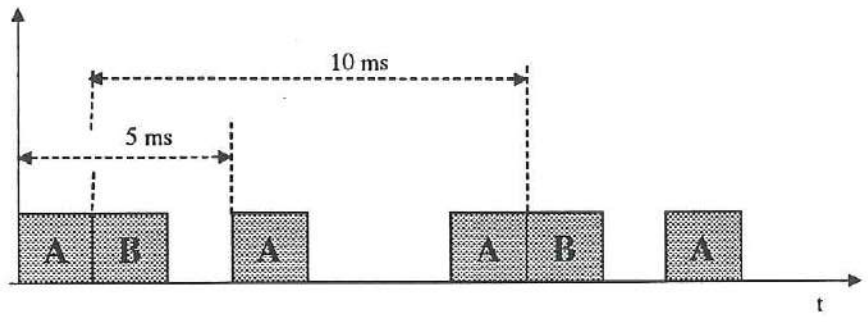


Figura 68: Restricción cíclica de tiempo para dos variables.

Nodos	2	4	8	16
125 kb/s	2	4	8	16
0.5 Mb/s	44,041	128,162	297,003	634,083
1 Mb/s	9,364	27,544	63,753	136,173

Tabla 47: Tiempos de respuesta (ms) para tabla más dos mensajes con restricción.

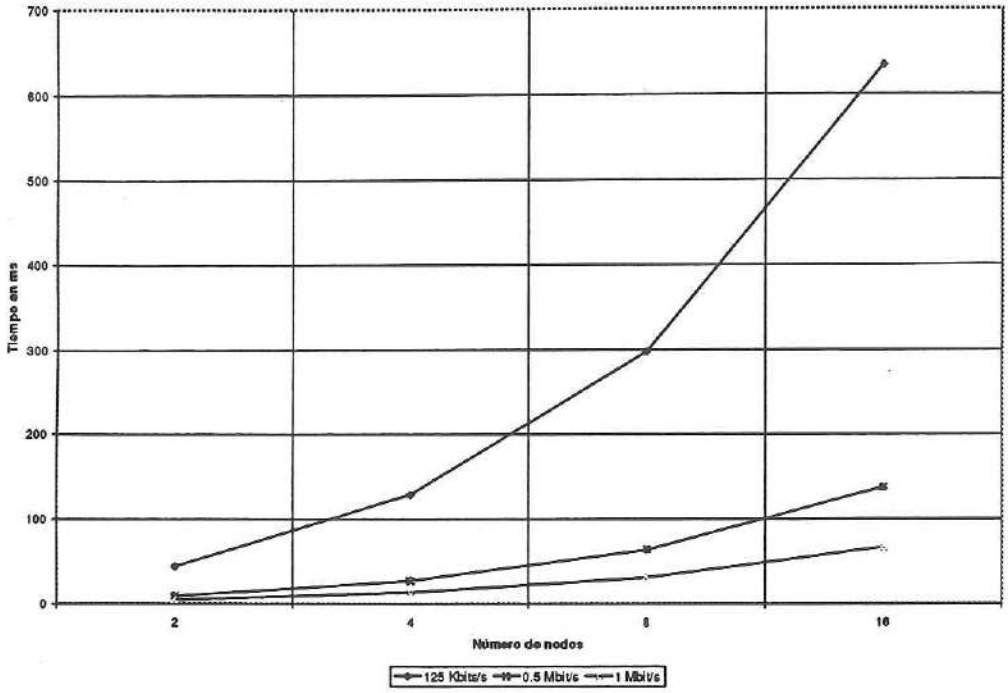


Figura 69: Tiempo máximo de respuesta con dos variables con restricción.

Los resultados presentados sólo cubren una pequeña parte de todas las posibilidades que se pueden encontrar en aplicaciones reales. Pero igualmente cualquier situación real puede ser representada y muestra la capacidad de la red de CCCs para efectuar las transferencias de datos requeridas.

Capítulo 6.

Prototipos de CCCs y pruebas de validación de la metodología propuesta.

En este capítulo se presentan dos modelos prototipo de CCC, diseñados y construidos en el Laboratorio de Equipos y Productos Industriales del Centro de Microelectrónica Aplicada de la ULPGC con el objeto de validar los resultados anteriores, demostrar la viabilidad de la metodología propuesta y comparar entre los modelos la influencia del tipo de estructura CPU / Controlador CAN, en el tiempo de retardo de los mensajes.

Para poder comparar la influencia en el tiempo de retardo debido a la forma de acceso de la CPU al controlador de bus CAN, los dos modelos han sido desarrollados con distintos mecanismos de acceso al mismo. Uno de ellos, direcciona el controlador como un periférico típico de cualquier arquitectura basada en la familia de microprocesadores 68000 de Motorola, por lo que accede directamente al mapa de memoria del circuito de interfaz, mientras que el otro, el que se basa en arquitectura PC, emplea el método convencional de acceso del bus ISA, que se realiza de forma indirecta.

También en el presente capítulo, en el apartado 6.3, se presentan dos ejemplos de aplicación a sistemas instalados y en funcionamiento, a los cuales se les aplica la metodología propuesta, y se comparan los resultados frente al sistema de integración convencional con que cuentan.

6.1 Prototipos de CCCs.

Antes de describir el diseño de los CCCs, se presenta el dispositivo utilizado como interfaz al bus CAN.

6.1.1 Circuito de acceso al bus.

Es necesario seleccionar el dispositivo que será utilizado de interfaz al bus CAN, de forma que cumpla con los requerimientos para la implantación de la metodología expuesta en el capítulo 5.

6.1.1.1 Selección del circuito de interfaz.

La selección del circuito de acceso al bus CAN debe realizarse entre dos grandes grupos; los que son simples controladores CAN (*stand-alone*), y los que forman parte de un sistema integrado microcontrolador - interfaz. En la Tabla 48 se muestra una relación de algunos de los dispositivos de interfaz más utilizados.

Fabricante	*Stand-Alone*	Microcontroladores de 8 bits	Microcontroladores de 16 / 32 bits
Hitachi	HCAN-1		H8/300H, SuperH
Intel	AN82527, AS82527		AN87C196CA/CB, AS87C196CB
Mitsubishi		M37630	
Motorola		MC68HC05XX, MC68HC08AZ, MC912D60/12BC32	MC68376
National Semiconductor		COP884BC, COP888EB	
NEC		78K/0	
Philips Semiconductors	SJA 1000 82C200	80592/98, XA-C3	XA-C3
SGS Thomson		ST6/7	ST10F167
Siemens	81C90/91	C505C, C515C	C167CR
Temic		TSC8051A11, TSC80251A3	
Texas Instruments			TMS 370E08D55

Tabla 48: Dispositivos de interfaz a bus CAN más utilizados.

Existen algunas diferencias entre los dispositivos *stand-alone* y los que están integrados dentro de un sistema microcontrolador, que se deben tener en cuenta. Los dispositivos que ya traen integrada la interfaz a bus CAN, resultan más baratos, no sólo por el precio de los mismos, sino porque el diseño de los circuitos impresos es mucho más económico, minimizando espacio y dinero. Respecto a los desarrollos software, los que ya tienen un microcontrolador están limitados a la capacidad de éstos, y la exportación de la aplicación está limitada a la familia de microcontrolador específica. En cambio los dispositivos de interfaz *stand-alone*, están diseñados para poder ser direccionados de muy diversas formas por los distintos tipos de microprocesadores presentes en el mercado, facilitando desarrollos software, que pueden incluso ser exportados a otros sistemas, aún con CPUs diferentes.

Respecto a la carga que representan para la CPU, los dispositivos que integran el controlador CAN realizan un tratamiento preliminar a los datos, liberando - a diferencia de los elementos *stand-alone* - a la CPU de tareas de control. Otro de los parámetros, que se diferencia en ambos tipos, es el tiempo de lectura / escritura. Éste es menor en los controladores integrados, pues acceden directamente al bus interno del dispositivo, normalmente diseñado para accesos rápidos.

En el presente trabajo hemos decidido utilizar elementos de interfaz *stand-alone*, por su gran flexibilidad de adaptación a las diversas arquitecturas de CPUs, para poder tener el control total de la gestión de los mensajes, y para poder exportar fácilmente todo el software realizado de una plataforma a otra. De esta manera podremos comparar las prestaciones del sistema bajo distintas configuraciones hardware, con aplicaciones software bajo sistemas operativos multitarea en tiempo real.

Dentro de los elementos de interfaz *stand-alone*, la selección ha sido realizada teniendo en cuenta la forma en que ésta planifica el envío de los mensajes de su cola de transmisión. El orden de envío de los mensajes de la cola debe ser realizado según su prioridad ya que ésta es la característica principal que tienen que cumplir las transferencias con restricciones de tiempo [TiBu229], y es una de las condiciones supuestas en los cálculos realizados en 5.12.3. Por esta razón se ha descartado el uso de controladores que no cumplen con esta característica, por ejemplo el 82C200 de PHILIPS, que envía los mensajes de acuerdo a una cola tipo FIFO (*Firts In First Out*), es decir los envía tal como son puestos en la cola y no según su prioridad.

En este trabajo se ha seleccionado el controlador CAN 82527 de INTEL, pues, aparte de reunir las características de transmisión de mensajes prioritarios, tiene uno de sus mensajes dedicado a la recepción con doble buffer de memoria, para permitir la llegada de dos mensajes consecutivos, y porque permite implantar cualquiera de los estándares de este tipo de bus. Otras de las razones para su selección son su amplia difusión, bajo coste y disponibilidad en el mercado.

6.1.1.2 Descripción del circuito de interfaz.

El controlador serie para bus CAN 82527 está diseñado para realizar transmisiones de **objetos de comunicación** entre nodos en configuración de maestro múltiple, realizando varias funciones adicionales, como por ejemplo: filtrado de mensajes, tratamiento de errores de transmisión, pedido de transmisión, etc., con el mínimo necesario de interacción con la CPU a la que se encuentre conectado.

Los **objetos de comunicación** son estructuras compuestas por un identificador unido a segmentos de control y datos. Los segmentos de control contienen toda la información necesaria para transferir el mensaje, y el segmento de datos contiene de 0 a 8 bytes en cada mensaje. Todos los objetos de comunicación están almacenados en la memoria RAM de cada controlador CAN en cada nodo del sistema. La Figura 70 muestra la estructura de un objeto mensaje de CAN.

El estado de una transmisión en curso, es monitorizado en el segmento de control del objeto de comunicación respectivo, dentro del nodo que la está originando. Si este nodo, mientras está transmitiendo, detecta algún error automáticamente inicializa una secuencia de retransmisión.

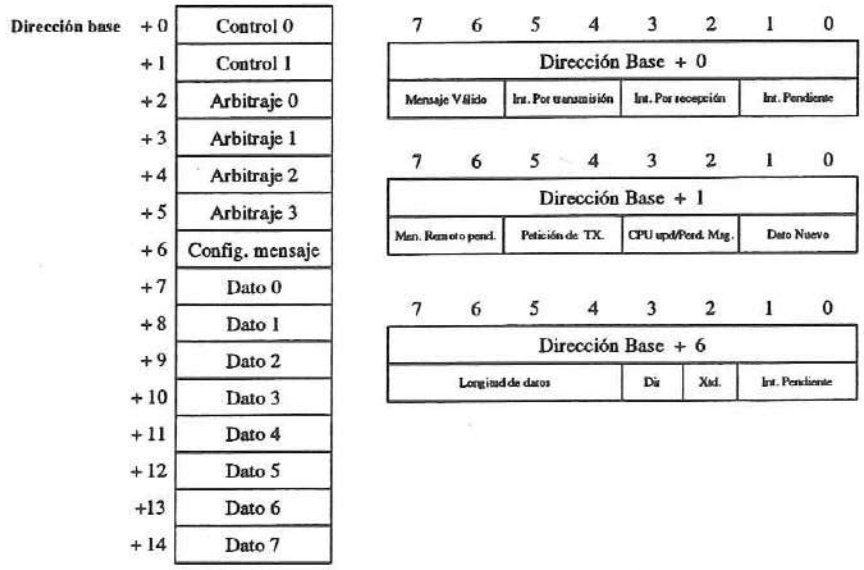


Figura 70: Estructura y partes principales de un objeto CAN.

Este tipo de controlador tiene una capacidad de almacenamiento de 15 objetos de 8 bytes de datos. Cada uno de los mensajes puede ser configurado tanto en transmisión como en recepción, excepto el último que sólo es de recepción. Idealmente un controlador CAN, para la versión estándar (11 bits de identificador, para direccionar hasta 2032 objetos), debería tener capacidad para tratar 2032 objetos, pero dado que ésto ocuparía una gran cantidad de memoria, normalmente los controladores CAN soportan un número inferior de objetos de comunicación. Actualmente existen tarjetas para acceso CAN para bus ISA, con procesadores integrados, que permiten recibir los 2032 objetos de comunicación que se pueden definir en una red de este tipo. Su utilización se justifica en sistemas con restricciones de tiempo muy exigentes, o en aplicaciones con un gran flujo de datos, donde es necesario dejar libre a la CPU la mayor cantidad de tiempo posible, para que tenga capacidad de proceso de todos los objetos. La Tabla 49 indica algunas tarjetas de este tipo.

Tipo	Fabricante	Procesador	Costo
PCI-CAN	National	80386EX	alto
CAN-AC1	esd gmbh	NEC V25+	medio
M26	mikro elektronik	MC68302	alto
ICAN?	JANZ computerag AG	MC68302	alto

Tabla 49: Tarjetas de interfaz a bus CAN para PC.

La Figura 71 muestra una forma típica de cómo el circuito de interfaz 82527 realiza el envío de sus mensajes objetos. En esta figura se muestra que el procesador *host* está poniendo en la cola de transmisión el objeto con identificador "1", dentro de la zona de memoria que le

corresponde. También se puede ver que el objeto con identificador "4" ya está puesto en dicha cola. Se supone que los objetos con identificadores 0, 2, 3, etc., no han sido puestos en la cola.

Los espacios de memoria, disponibles para cada objeto, están realizados físicamente como memoria de doble puerto, que es compartida entre la CPU y el procesador de interfaz. El procesador de interfaz intentará transmitir el objeto "1", que es el que tiene el identificador de mayor prioridad, cuando el bus se encuentre nuevamente libre; luego de transmitirlo intentará la transmisión del objeto "4". No existe una cola de mensajes para un identificador determinado: en la Figura 71, si el objeto "1" se empieza a transmitir cuando otro objeto, con el mismo identificador, es puesto en la cola, entonces, el mensaje es sobrescrito y destruido. Es muy importante de tener ésto en consideración, ya que implica que los objetos que son puestos en la cola periódicamente, tengan un tiempo límite para su transmisión; en otras palabras un objeto determinado debe ser transmitido antes de que el mensaje vuelva a ser puesto en la cola en el siguiente periodo. Esto significa que, los tiempos límites de los objetos deben ser más pequeños que los periodos de los mismos.

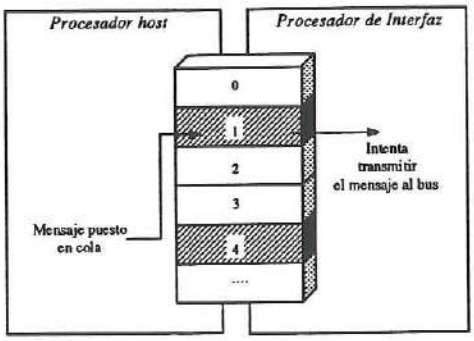


Figura 71: Envío de objetos en una interfaz para bus CAN 82527.

La estructura interna, del circuito de interfaz, se muestra en la Figura 72.

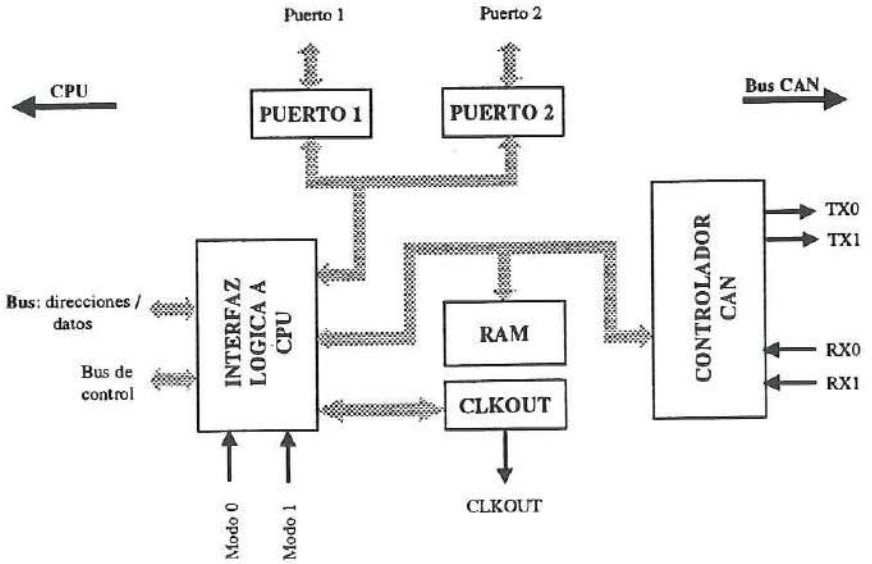


Figura 72: Diagrama de bloques del circuito de interfaz.

El controlador CAN se encarga de manejar el flujo de datos entre la memoria RAM (que tiene los datos en paralelo), y la línea serie del bus. También es el encargado de gestionar la lógica del *transceiver* (con las líneas TX0, TX1, RX0 y RX1), la gestión lógica de los errores y de los objetos.

La memoria RAM, que se direcciona desde la posición 00H hasta la FFH está compartida entre el bus y la interfaz lógica a la CPU, por medio del controlador CAN.

La interfaz lógica a la CPU proporciona los mecanismos de interconexión al procesador *host* de manera flexible, es así que cuenta con 4 formas posibles, resultado de la combinación de las entradas (Modo 0) y (Modo 1) respectivamente. La forma seleccionada, dependerá de las características de la CPU. En el caso de arquitectura PC utilizamos el tipo multiplexado – tipo 2 -, de direcciones y datos; y para la CPU propia el tipo no multiplexado – tipo 3-, el resumen de los modos de interconexión se muestran en la Tabla 50.

Tipo	Entrada modo 0	Entrada modo 1	Modo de funcionamiento
0	0	0	8-bit multiplexado
1	0	1	16-bit multiplexado
2	1	0	8-bit multiplexado
3	1	1	8-bit no multiplexado

Tabla 50: Tipos de interconexión del 82517.

La interfaz tiene disponibles dos puertos para entradas / salidas de 8 bits cada uno. De acuerdo al modo de funcionamiento seleccionado se podrá acceder a los dos o sólo a un puerto. En nuestro caso el CCC que utiliza modo 2 tiene posibilidad de acceso a ambos puertos, mientras que el que hace uso de modo de acceso 3 sólo dispone de un puerto.

6.1.1.3 Cálculo de los parámetros del tiempo de un bit.

El estándar CAN (parte 8 de [CAN2.0]), indica solamente los límites permitidos para la partición del tiempo asignado a un bit (para sincronización, muestreo del valor del bit, etc.), dejando que el usuario realice el cálculo, de acuerdo a sus requerimientos particulares; por ejemplo, garantizar máxima distancia de transmisión o seguridad en la toma del valor del bit (esto último en ambientes muy ruidosos).

Dado que este cálculo es muy importante, para el correcto funcionamiento del bus, en este apartado explicamos la forma en que ha sido realizado, presentando además las pruebas realizadas, con los parámetros seleccionados, para la verificación del funcionamiento correcto del bus. Previamente, en la Figura 73, se muestra la distribución del mapa de direcciones del dispositivo indicando las principales funciones de algunos registros importantes.

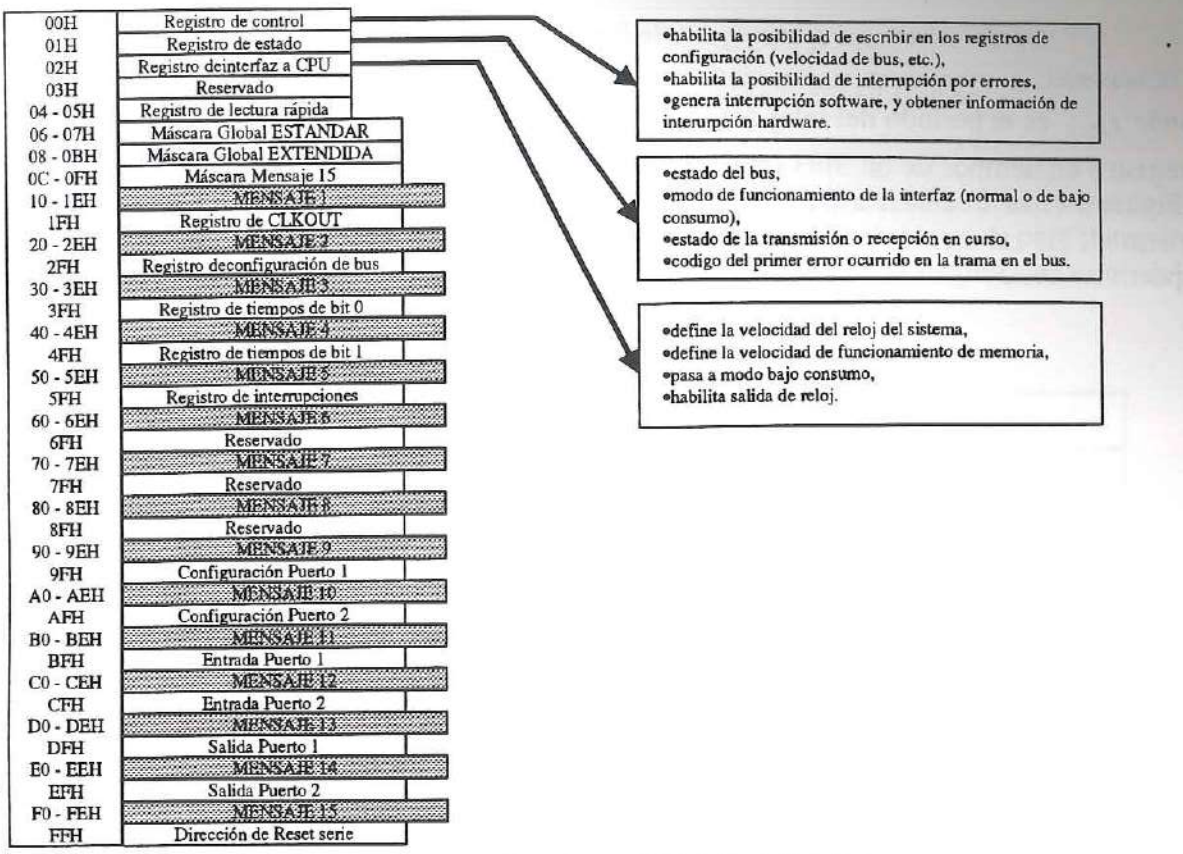


Figura 73: Mapa de direcciones del circuito de interfaz.

Respecto a la división del tiempo de un bit en CAN, ésta se realiza tal como muestra la Figura 74.

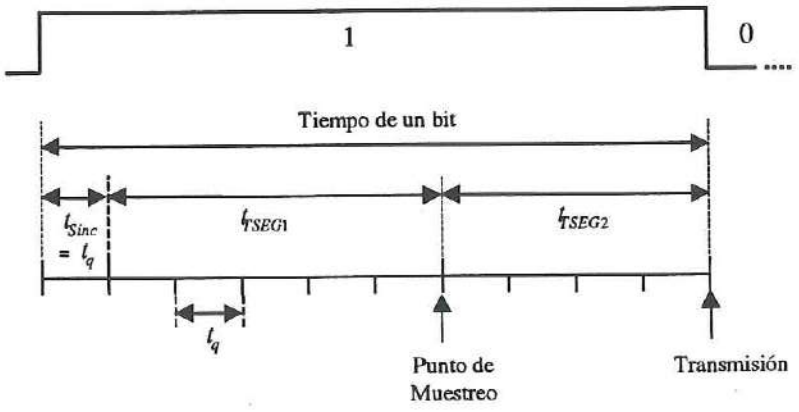


Figura 74: División del tiempo de un bit.

El tiempo t_q es una unidad de tiempo fija obtenida a partir del reloj del sistema que se calcula de la manera siguiente:

$$t_q = t_{SCLK} \times (BRP + 1)$$ Ecuación 20

Donde t_{SCLK} es el periodo del reloj del sistema y BRP es el factor de escala de t_q indicado en el registro de tiempos de bit 3FH (puede variar entre 0 – 63). Los otros tiempos, indicados en la Figura 74, se obtienen a partir de los factores TSEG1 y TSEG2 que forman parte del registro 4FH, tal como indica la Figura 75, con la Ecuación 21 y la Ecuación 22 respectivamente.

Figura 75: Registros de configuración de tiempos de un bit.

$$t_{TSEG1} = (TSEG1 + 1) \times t_q$$ Ecuación 21

$$t_{TSEG2} = (TSEG2 + 1) \times t_q$$ Ecuación 22

El tiempo indicado en la Ecuación 21 debe tener en cuenta el retardo físico de propagación de los bits en la red, más los retardos debidos al comparador de entrada y al driver de salida de las dos interfaces en comunicación. Opcionalmente, cuando el valor del bit "Spl" – bit 7 – del registro 4FH tiene valor 1, se toman tres muestras del valor del bit – asignándole valor 1 al bit "Spl" del registro 4FH - (empleado en ambientes con mucho ruido) en lugar de una sola muestra. En este caso las dos muestras adicionales incrementan el valor de este tiempo en $2 \times t_q$. El valor del dato, en el caso en que las tres muestras no sean iguales, se toma igual a la muestra que más se repite.

Con las consideraciones anteriores, las restricciones impuestas por el estándar y los requerimientos particulares de la red, se pueden calcular los valores de los parámetros de los registros 3FH y 4FH. Las restricciones impuestas por el estándar son tres: la primera indica que el tiempo de un bit debe ser dividido al menos en ocho partes; las otras dos están indicadas en la Ecuación 23 y la Ecuación 24.

$$t_{TSEG2} \geq 2 \times t_q$$ Ecuación 23

$$t_{TSEG1} \geq 3 \times t_q$$ Ecuación 24

De la Ecuación 20 a la Ecuación 22 podemos deducir la frecuencia a la que funcionará el bus, cuya expresión se indica en la Ecuación 25.

$$f_{CAN} = \frac{1}{t_{Sinc} + t_{SEG1} + t_{SEG2}} = \frac{1}{t_q + (TSEG1+1)\times t_q + (TSEG2+1)\times t_q} = \frac{1}{(3+TSEG1+TEG2)\times t_q} \quad \text{Ecuación 25}$$

Todos los parámetros de tiempo del bus, calculados a una frecuencia del reloj del sistema de 8 Mhz, se muestran en la Tabla 51. En la tabla se muestran los valores calculados para cumplir tanto el requerimiento de distancia máxima, como el de seguridad en la lectura del valor del bit.

Frecuencia del reloj del sistema (8 Mhz)

BRP	t_q (µs)	TSEG1	TSEG2	Muestras del bit	f_{CAN}	%dismi. dist.
0	0.125	4	1	1	1 Mb/s	-
0	0.125	4	1	3	0.8 Mb/s	-
0	0.125	12	1	1	500 kb/s	-
0	0.125	10	1	3	500 kb/s	15%
1	0.250	12	1	1	250 kb/s	-
1	0.250	10	1	3	250 kb/s	15%
3	0.375	12	1	1	125 kb/s	-
3	0.375	10	1	3	125 kb/s	15%

Tabla 51: Parámetros de los tiempos de un bit.

En todos los casos se ha considerado que las muestras del valor del bit, se toman en el instante más tarde que permite el estándar. Ésto se hace con el fin de obtener el funcionamiento del bus con la mayor longitud posible. Recordamos que en este bus todos los nodos deben enterarse al mismo tiempo del valor del bit – es decir realizar la toma de la muestra en un momento determinado – y por consiguiente si tomamos las muestras casi al final del bit, podemos aprovechar la máxima longitud permitida por los retardos de propagación de la señal en el medio físico.

En la Tabla 51, fila sombreada, se puede observar que con una frecuencia de reloj del sistema de 8 MHz (caso de este trabajo), no se puede obtener una frecuencia de bus de 1 Mb/s, ya que al incrementar dos periodos de muestreo el tiempo total de bit también se incrementa, pudiéndose obtener una velocidad máxima de sólo 0.8 Mb/s. La Tabla 52 muestra los valores de los parámetros considerando la máxima frecuencia de trabajo del sistema 10 MHz.

Frecuencia del reloj del sistema (10 Mhz)

BRP	t_q (µs)	TSEG1	TSEG2	Muestras del bit	f_{CAN}	%dismi. dist.
0	0.1	6	1	1	1 Mb/s	-
0	0.1	4	1	3	1 Mb/s	15%
1	0.2	6	1	1	500 kb/s	-
1	0.2	4	1	3	500 kb/s	15%
3	0.4	6	1	1	250 kb/s	-
3	0.4	4	1	3	250 kb/s	15%
4	0.5	12	1	1	125 kb/s	-
4	0.5	10	1	3	125 kb/s	15%

Tabla 52: Parámetros de los tiempos de un bit.

En todos los casos en los que manteniendo la velocidad de bus se requiera incremento en la seguridad de lectura del valor de bit (caso de ambientes muy ruidosos), es recomendable tomar tres muestras del valor del bit. Los retardos añadidos llevan al compromiso de disminuir la longitud del bus; en nuestro caso este decremento vale aproximadamente 15%. Este porcentaje se estima fácilmente, ya que en todos los casos (de aumento de 2 muestras del valor del bit) el parámetro TSEG1 debe decrementarse en dos unidades, tomándose la primera muestra dos unidades (t_q) de tiempo antes.

6.1.2 Gestión de los intercambios en los nodos.

A continuación se describe cómo se realiza la gestión de los envíos y recepciones de todas las variables definidas en cada nodo, según se describió en el apartado 5.10.4, para su implementación en las dos arquitecturas de CCCs, realizadas con interfaz a bus CAN con el 82527.

6.1.2.1 Diagrama de flujo del proceso de envío de mensajes de operación y sin restricciones.

En este apartado se presenta el diagrama de flujo de este proceso teniendo también en cuenta la parte que es realizada por el hardware del circuito de interfaz. El diagrama de flujo de este proceso se muestra en la Figura 76. Los parámetros de entrada son el número de segmentos de tabla, o conexiones que debe realizar, para transmitir todas las variables de los elementos integrados; y el tiempo de retardo de tabla (t) que calcula el CCCP para cumplir todas las restricciones de tiempo del sistema (tal como se explicó en el capítulo 5).

El algoritmo de envío implantado inicializa un reloj, indicado como TMR1, y luego procede a verificar si ha variado el primer segmento que compone la tabla. En caso de que el objeto haya cambiado se procede a su envío a la red, verificando y actuando en cada caso sobre el estado del semáforo. Luego de verificar el envío, se procede a comprobar si existe algún mensaje de operación a transmitir, si éste existe se transmite seguidamente; en caso contrario se procede como antes a verificar si el segundo segmento ha variado.

Terminada la verificación de todos los elementos de la tabla, se espera a que el reloj alcance el tiempo de espera de transmisión de cada nodo, y se procede de nuevo como se explicó antes. De esta forma, los objetos sólo son transmitidos cuando alguna de sus variables cambia. En caso contrario no se transmite, disminuyendo el tráfico en la red.

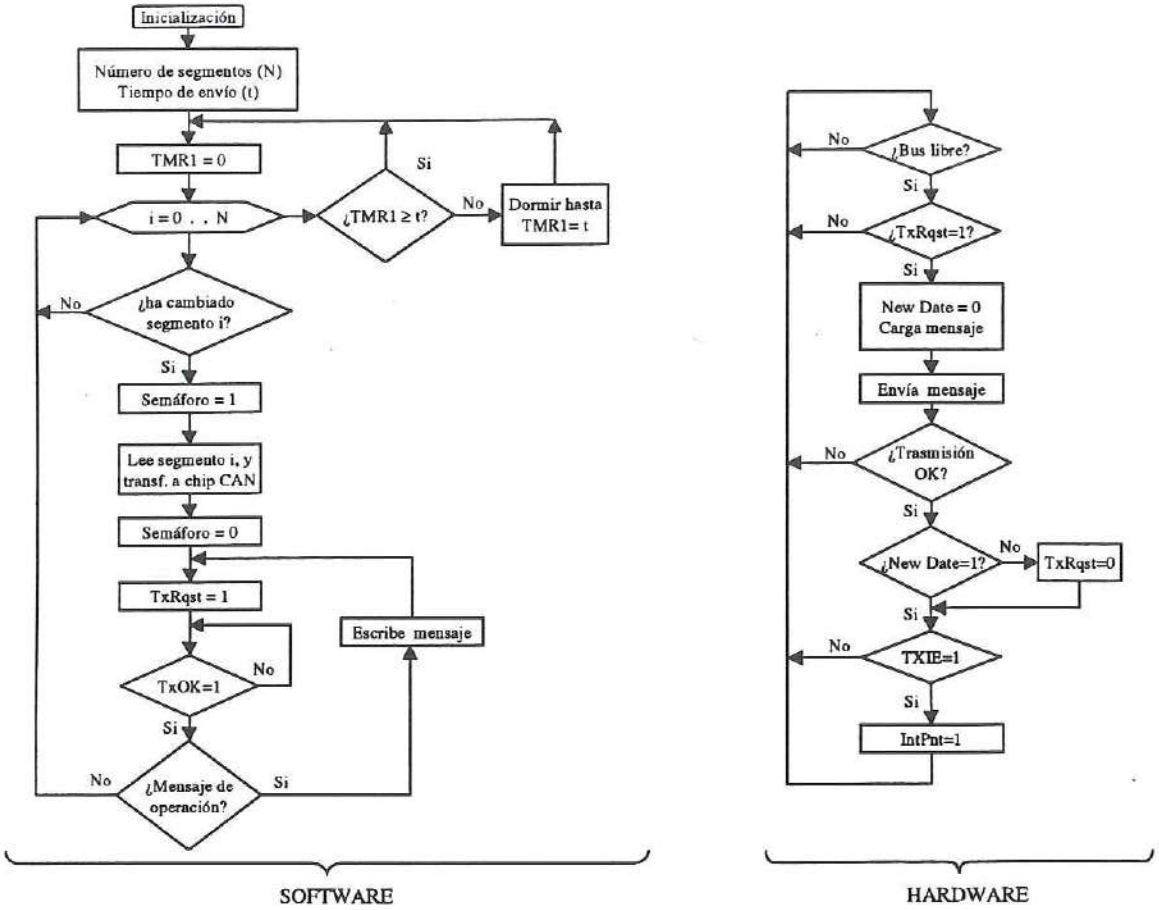


Figura 76: Diagrama de flujo de envío de mensajes de operación y sin restricciones.

6.1.2.2 Proceso de recepción de mensajes de operación y sin restricciones.

Este proceso es el encargado de recibir los mensajes (ordenes y consignas), relacionadas con los elementos integrados. El diagrama de flujo correspondiente se muestra en la Figura 77.

Según se ha indicado en el apartado 5.10.4, cuando el circuito de interfaz recibe un objeto, destinado al nodo, mediante una breve rutina de interrupción copia su valor en una cola de datos definida para tal fin. Esta rutina envía una señal para avisar al proceso de recepción que hay un nuevo dato en la cola.

Si existe más de un mensaje en ésta lee el contador de mensajes, y procede a leer todos los elementos de la estructura, en la dirección indicada por el puntero de inicio de cola. Luego de la lectura incrementa el valor del puntero y decrementa el valor del contador de mensajes hasta que no quede ninguno. En este punto el proceso queda inactivo hasta nuevo aviso por parte de la rutina de interrupción. Si el mensaje recibido es de operación procede según la correspondiente rutina de atención específica, en caso contrario verifica el estado del semáforo y procede a escribir los datos en la zona de tabla correspondiente.

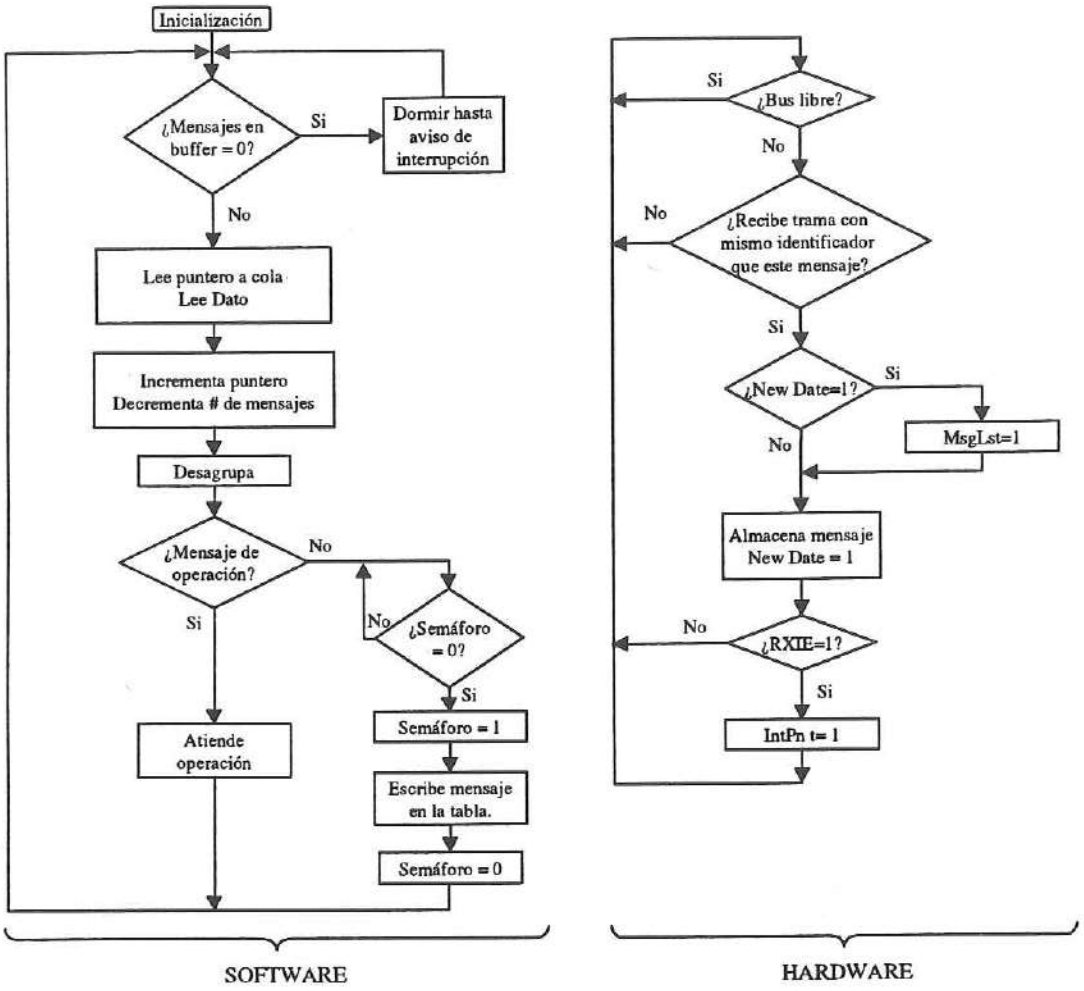


Figura 77: Diagrama de flujo del proceso de recepción de datos de operación y sin restricciones.

6.1.2.3 Proceso de envío de variables cíclicas.

Algunas variables del sistema pueden presentar restricciones, en cuanto a envío a consistencia temporal o espacial, o respecto a envíos de forma cíclica. En el caso de las variables con restricciones de tiempo, pero que se presentan de manera esporádica, estas también son enviadas por medio de este mismo proceso.

Para las variables cíclicas el algoritmo de planificación de envío agrupa a estas variables de acuerdo al tiempo cíclico que tengan definido, tal como se explicó en 5.13.4. El algoritmo para realizar estas transferencias se muestra en la Figura 78.

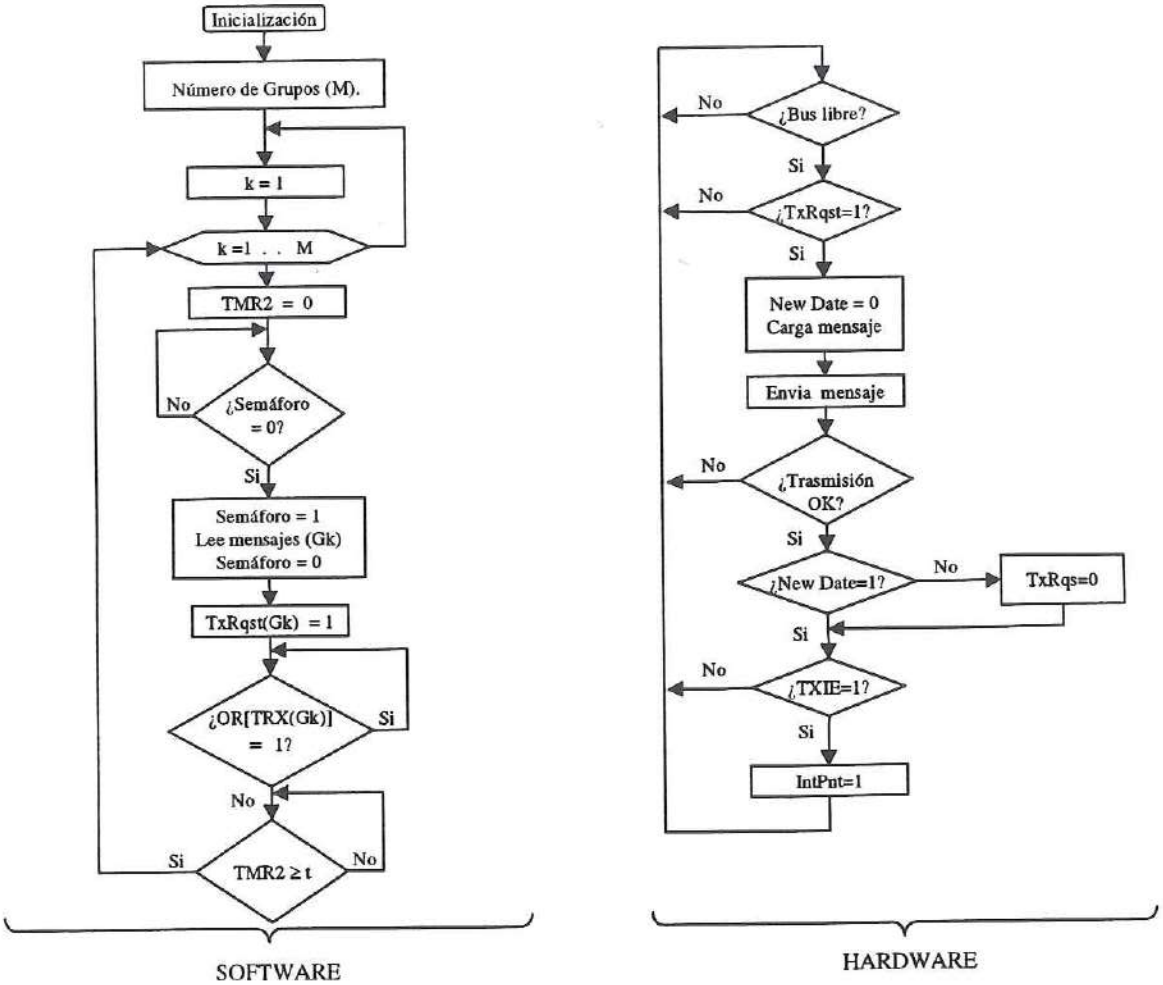


Figura 78: Diagrama de flujo de envío de variables cíclicas.

6.1.2.4 Proceso de recepción de variables con restricciones.

Para completar el tratamiento de las variables con restricciones, en este apartado describimos brevemente cómo se realiza su gestión. En la Figura 79 se muestra el diagrama de flujo del proceso responsable.

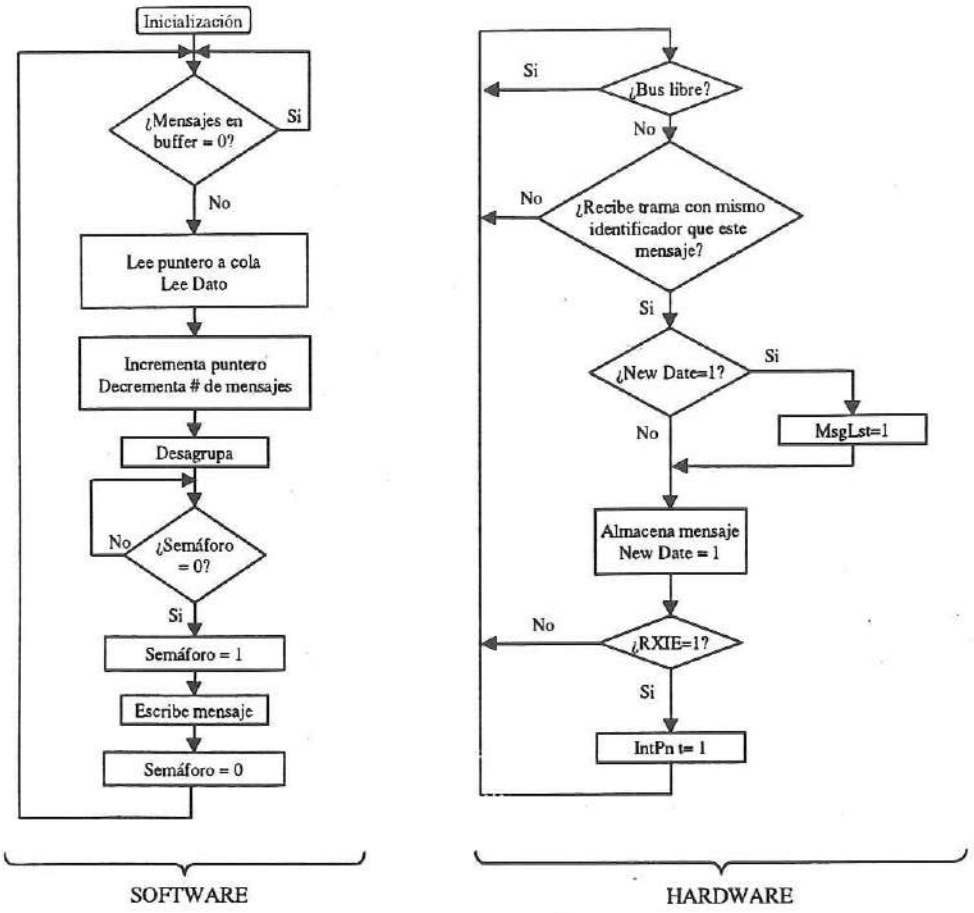


Figura 79: Diagrama de flujo de la recepción de variables con restricciones.

6.1.3 CCC basado en bus ISA.

Se ha diseñado un CCC basado en la arquitectura PC sobre bus ISA, por la gran difusión que tienen los ordenadores basados en esta arquitectura, y porque la mayoría de las veces el sistema supervisor general estará realizado con un PC.

Dadas las características del medio donde normalmente estarán instalados los CCCs, el hardware debe reunir algunos requisitos indispensables para el correcto funcionamiento del sistema. Estos requisitos se resumen a continuación:

- CPU monotarjeta (386-SX ó 486-DX2,DX4, etc).
- Tarjeta CAN sobre bus ISA o PC104.
- Tarjeta de red (opcional)
- Backplane de 3 ó 4 slots ISA.
- Chasis industrial compacto.
- Tarjeta de E/S Analógicas y Digitales (opcional).
- Tarjeta módem o de red (opcional).
- Tarjeta emuladora de disco de estado sólido.
- Fuente de Alimentación según aplicaciones.

- Posibilidad de incorporar diversos Sistemas Operativos en tiempo real, OS900, QNX, etc.

La Figura 80 muestra un sistema con este tipo de arquitectura.

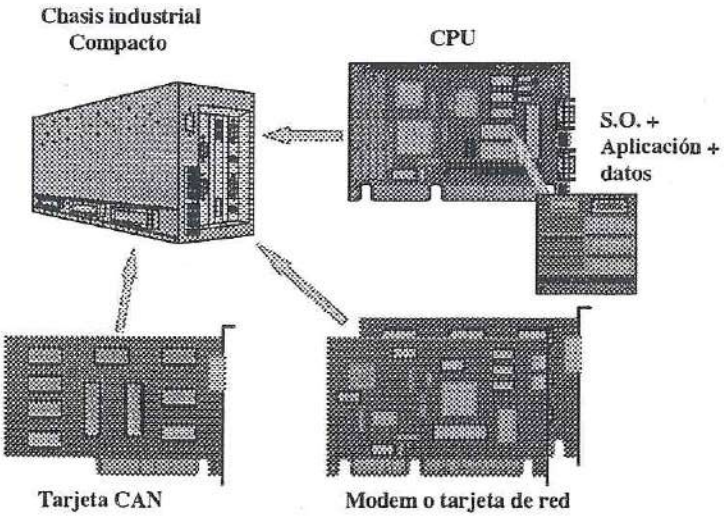


Figura 80: CCC con bus ISA

Para permitir que la tarjeta CAN diseñada sea compatible con todos los sistemas PC y tarjetas periféricas, se realizó un diseño incluyendo un circuito 8255 de interfaz al bus ISA, según el diagrama de bloques mostrado en la Figura 81. En la Figura 82 se muestra una foto de la interfaz realizada, y en la Figura 83 se muestra el montaje sobre el chasis industrial. En primer plano se observa la CPU industrial monotarjeta.

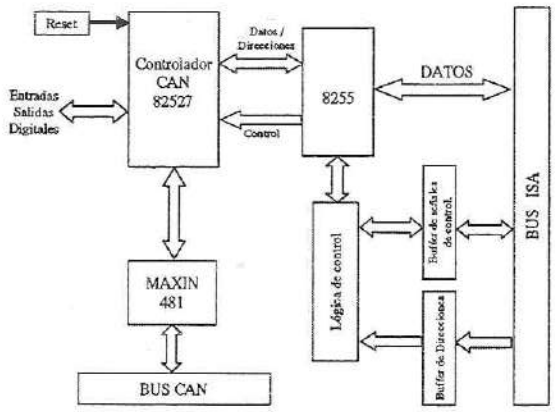


Figura 81: Diagrama a bloques del sistema con arquitectura PC.

El acceso de la tarjeta de interfaz al medio ha sido realizado con el circuito MAX481 que es un *transceiver* para RS-485 que permite transferencias de hasta 2.5 Mb/s. Los dos puertos libres, del circuito de interfaz, pueden ser utilizados como puertos discretos de hasta 16 entradas / salidas.

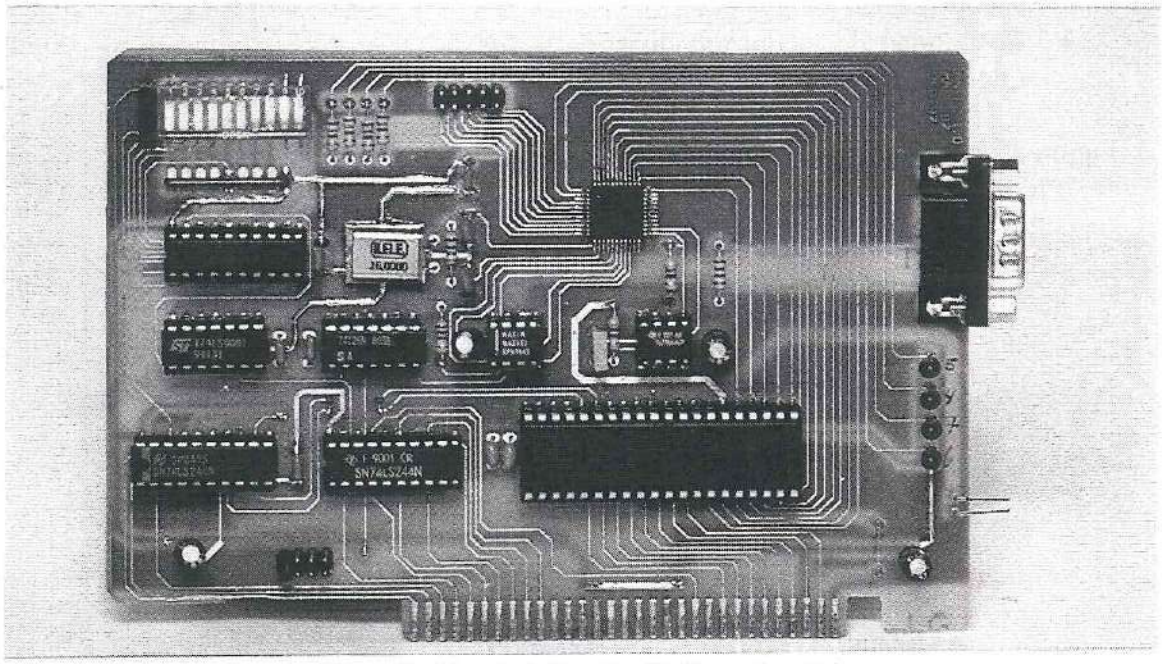


Figura 82: Foto de la interfaz CAN para bus ISA.

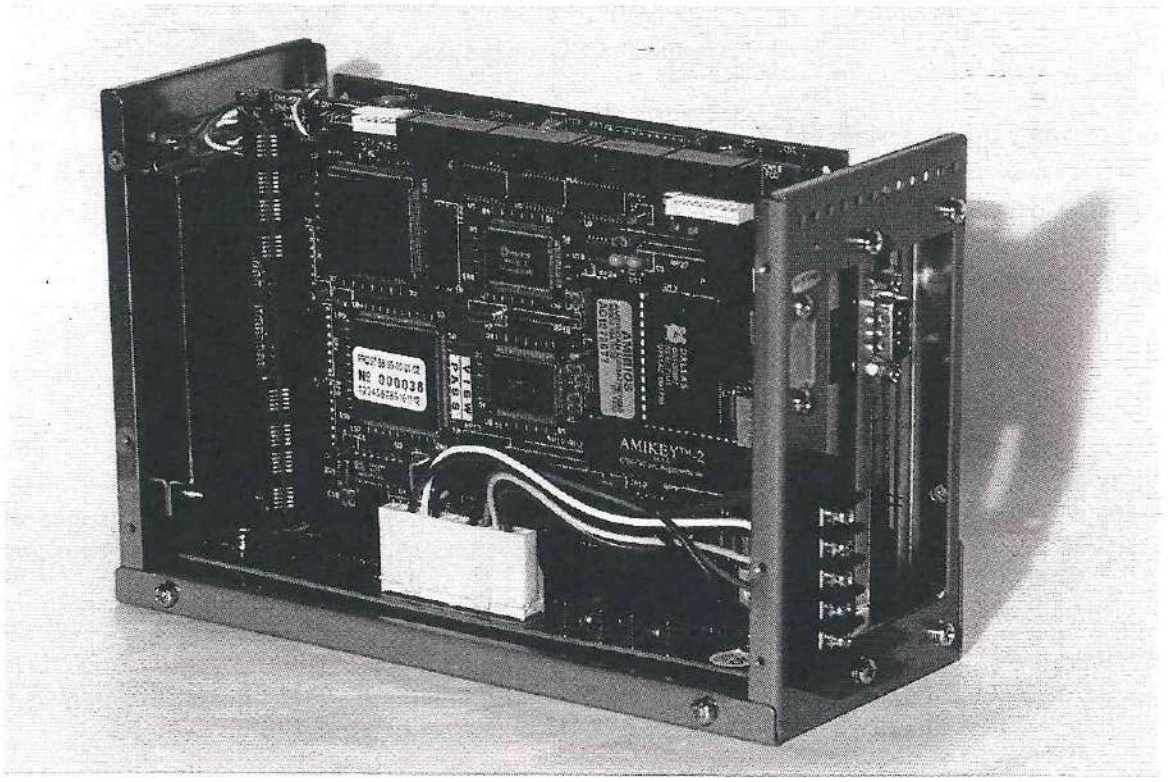


Figura 83: Montaje de interfaz y CPU monotarjeta en chasis industrial.

Uno de los parámetros importantes, según el manual del 82527, que debe tenerse en consideración es el tiempo de duración de la señal de reinicialización del dispositivo, la cual se debe garantizar que permanezca a cero por un tiempo de al menos 1 ms. Para la correcta inicialización del dispositivo, este tiempo debe ser considerado a partir del instante en que la tensión de alimentación alcance su valor nominal. Para realizar esta función se ha utilizado un dispositivo, el LM7705, dedicado para tal fin.

El acceso al bus ISA se realiza mediante un circuito 8255 que tiene 4 puertos (A, B, C y control) que permite acceder, mediante gestión de software, a los 256 registros del circuito de interfaz. Los modos de funcionamiento de este circuito se indican en la Tabla 53.

A1	A0	*RD	*WR	*CS	Operación de lectura	
					De	a
0	0	0	1	0	Puerto A	Bus de datos
0	1	0	1	0	Puerto B	Bus de datos
1	0	0	1	0	Puerto C	Bus de datos
					Operación de escritura	
0	0	1	0	0	Bus de datos	Puerto A
0	1	1	0	0	Bus de datos	Puerto B
1	0	1	0	0	Bus de datos	Puerto C
1	1	1	0	0	Bus de datos	Control
					Desactivado	
X	X	X	X	1	Bus de datos en triestado	
1	1	0	1	0	Ilegal	
X	X	1	1	0	Bus de datos en triestado	

Tabla 53: Modos de funcionamiento del 8255.

Para acceder a este circuito la lógica de control tiene 8 microinterruptores que permiten seleccionar la dirección base 0 y 3FC en hexadecimal.

Respecto al sistema operativo, se ha utilizado el OS9000 que es un sistema multitarea en tiempo real.

6.1.4 CCC con CPU 68000 de Motorola.

Para poder comparar los tiempos de retardo hardware de distintas arquitecturas, se ha realizado una segunda versión del CCC, basada en un microprocesador de Motorola (el 68000). De esta manera se obtiene un CCC compacto que puede ser utilizado dentro de los entornos industriales; presentando buena fiabilidad (no tiene partes móviles), baja corriente de alimentación (300 mA el sistema completo a 5 V) y ocupando poco espacio. Otra de las ventajas significativas es el bajo coste del sistema.

Se han descartado los microcontroladores, pues uno de los requerimientos de los CCCs, según se vio en el capítulo 5, es la capacidad de soportar un sistema operativo multitarea en tiempo real. En este caso también se añade la restricción de que todo el sistema pueda ser instalado en memoria no volátil. Este tipo de CCC está pensado también para los casos de sistemas con condiciones industriales exigentes, donde no se necesita monitor.

La CPU realizada tiene disponibles, a parte del puerto CAN, dos puertos serie que pueden ser utilizados como RS-232 o RS-485. Para hacer el sistema compacto y con flexibilidad a cambios, la lógica de control del sistema ha sido instalada en dispositivos lógicos programables tipo GAL 20V8.

El diagrama de bloques de todo el sistema se muestra en la Figura 84. La Figura 85 muestran la tarjeta de interfaz a bus CAN, la Figura 86 la CPU diseñada, y en la Figura 87 se muestra

una foto del sistema completo dentro de su chasis que incluye fuente de alimentación y *backplane*. La CPU se ha diseñado con un bus local para dar flexibilidad al sistema, en cuanto a la posibilidad de aceptar diferentes tarjetas de buses industriales. Por esta razón la tarjeta de interfaz, a bus CAN, ha sido desarrollada como tarjeta de expansión de este bus.

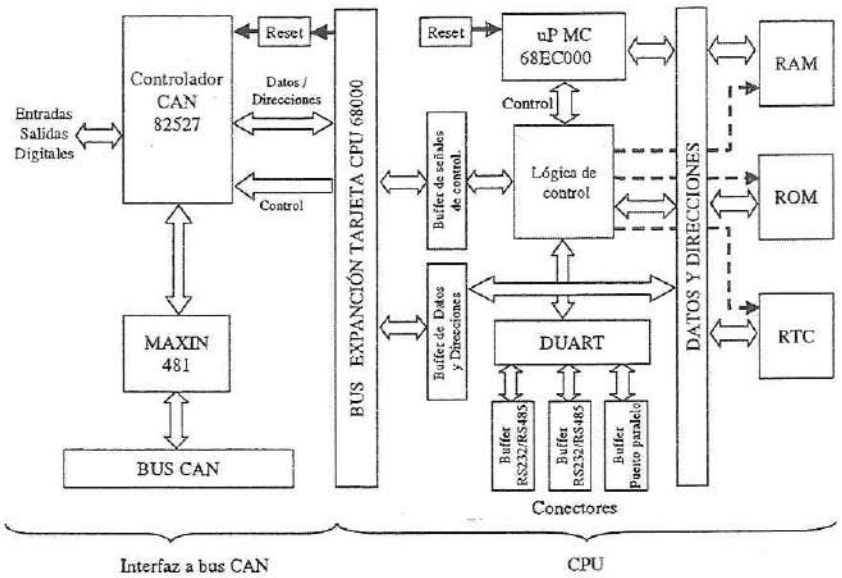


Figura 84: Diagrama de bloques del sistema con CPU 68000.

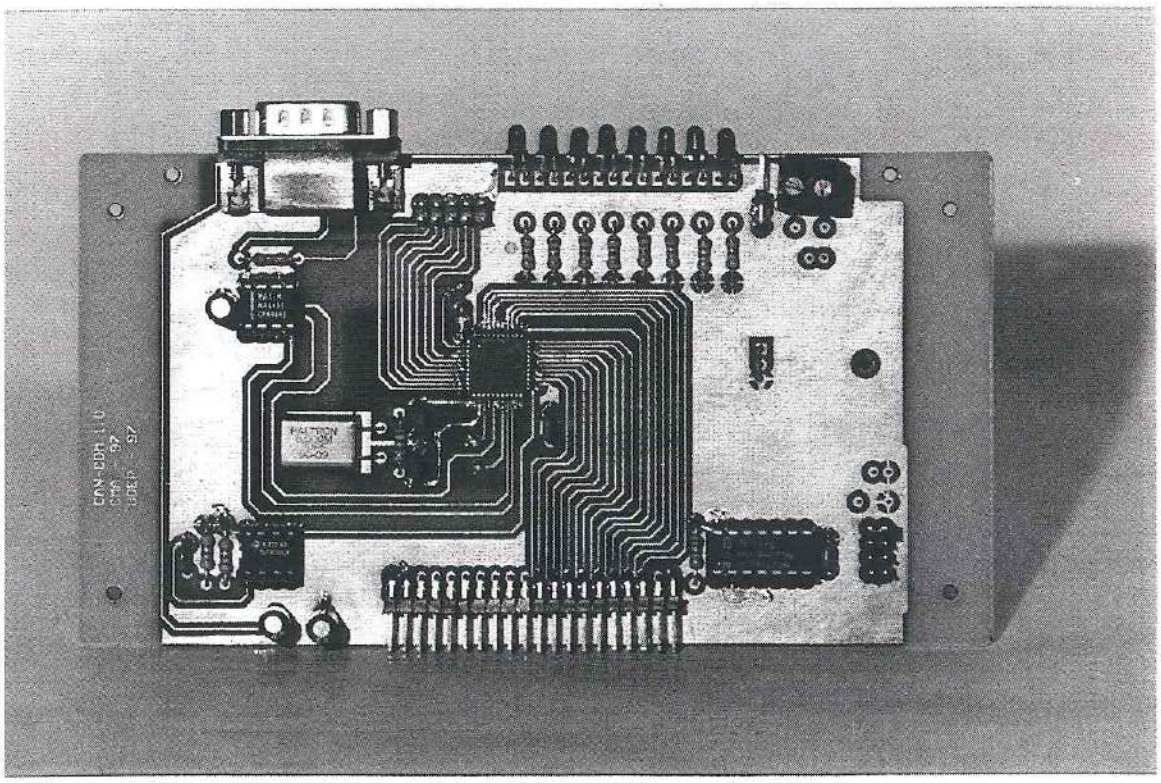


Figura 85: Foto de la tarjeta de interfaz a bus CAN para CPU 68000.

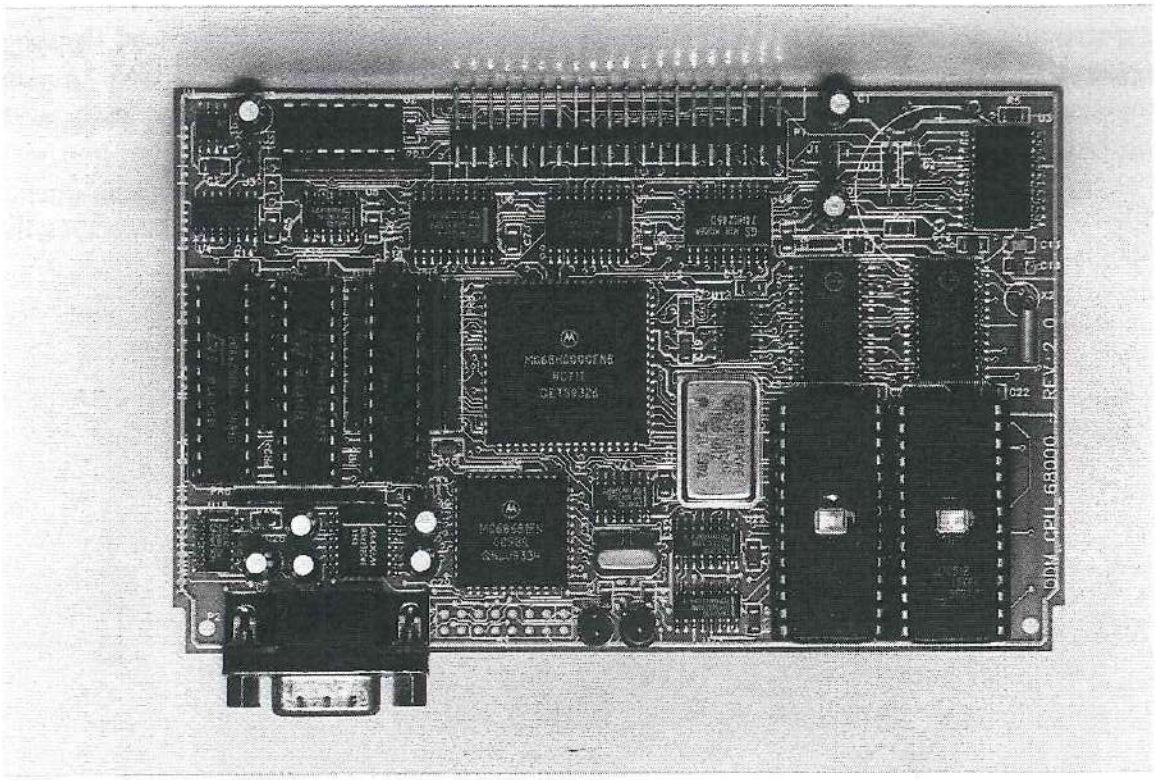


Figura 86: Foto de la CPU 68000.

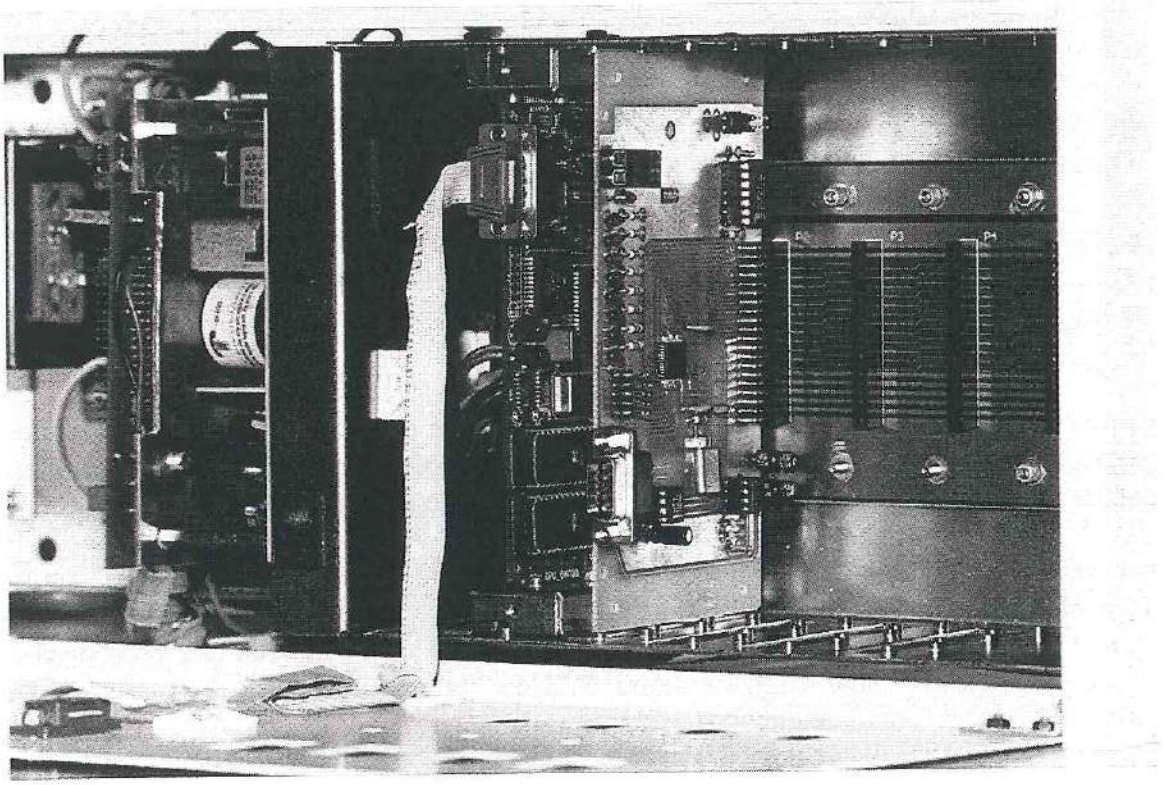


Figura 87: Foto del sistema completo CPU / tarjeta de interfaz CAN.

Respecto al sistema operativo, se ha utilizado el sistema operativo multitarea en tiempo real OS9, que permite su implantación, en versión compacta, directamente en EPROM. Una de las ventajas de este sistema operativo es la facilidad de importar los programas realizados en OS9000.

En la Figura 88 se muestra una foto comparativa de entre los dos CCCs antes mencionados en su configuración más simple.

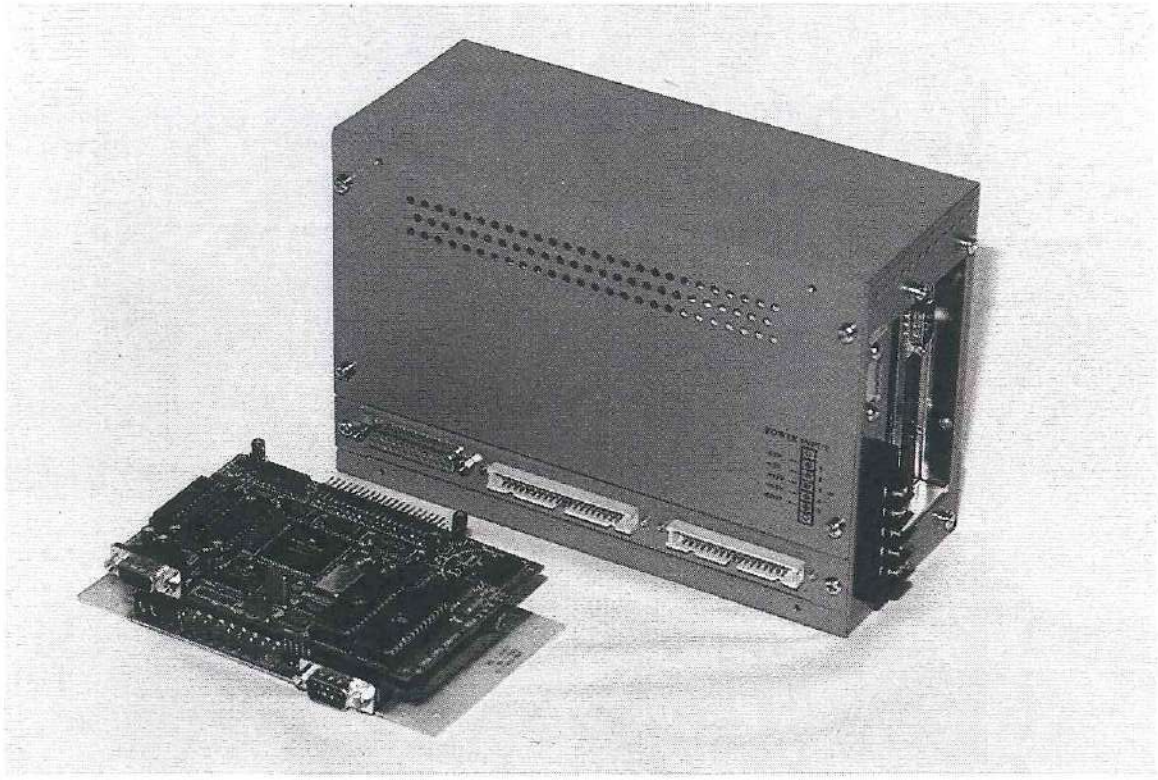


Figura 88: Comparación entre los dos CCCs.

6.2 Pruebas de validación.

En esta sección se describirán las pruebas realizadas para comprobar y validar los resultados presentados en el capítulo 5, y para comparar las dos plataformas hardware utilizadas. En todas las pruebas se ha utilizado una red montada tal como indica la Figura 89 .

Las estaciones 1 y 2 simulan los nodos que originan el tráfico correspondiente a la situación bajo análisis. El flujo de datos, emitido por un nodo, se simula utilizando uno de los 14 objetos de transmisión disponibles en la tarjeta de interfaz. Se utilizan dos estaciones pues cada una sólo cuenta con 14 objetos de transmisión, y como el sistema propuesto define 16 nodos, se necesitan dos estaciones para simular toda la red.

El nodo 3, que actúa como CCCP, indica a los otros nodos el momento de iniciar la transmisión, y calcula el tiempo total de transferencia de todos los mensajes.

Para medir este tiempo, en el momento de producirse la primera interrupción por recepción arranca un temporizador, que termina su cuenta cuando llega el último mensaje transmitido. El tiempo de transmisión de todos los mensajes se calcula luego sumándole el tiempo de transmisión del primer mensaje indicado en la Figura 89 como "Tiempo TX1" (pues en el momento de producirse la primera interrupción, cuando se arranca el temporizador, no se ha tenido en cuenta el tiempo que se tarda en transmitirlo).

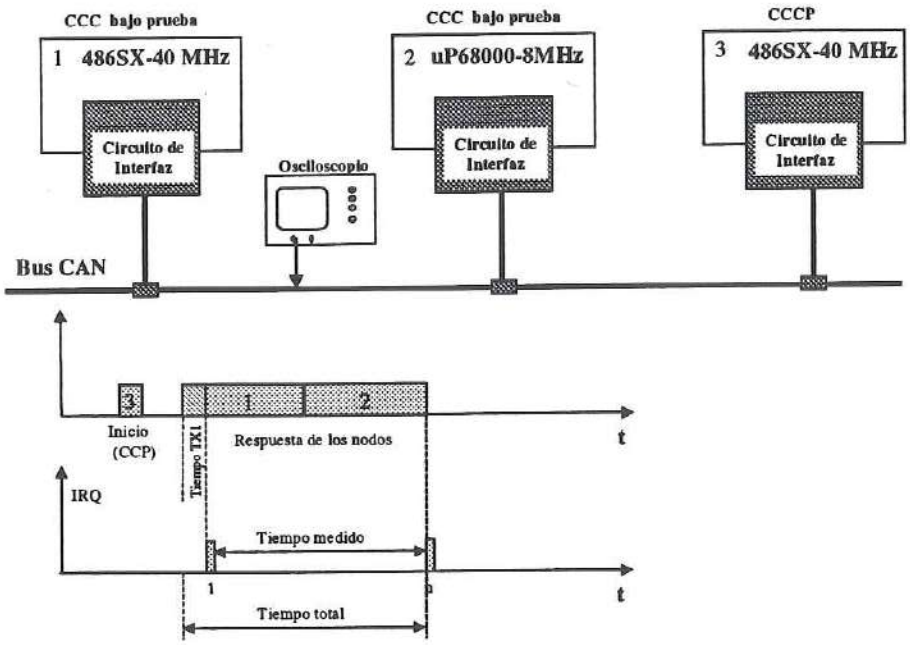


Figura 89: Red de pruebas utilizada.

Como comprobación adicional, medimos el tiempo haciendo uso de un osciloscopio, en modo barrido único, que empieza a muestrear con el flanco de bajada de la interrupción en el CCCP.

Como parte de las comprobaciones se han realizado las pruebas de retardo del software y de velocidad de funcionamiento máxima de bus a dos de los CCCs en arquitectura PC, uno de ellos en rack industrial como el mostrado en la Figura 80. Los resultados se presentan en el apartado siguiente.

6.2.1 Medición de retardo del software y de velocidad máxima de funcionamiento.

Estas pruebas han sido realizadas en dos CCCs diversos: el CCC1 basado en un PC industrial con procesador 486SX40, y el CCC2 basado en un PC 486 DX33.

Medición del tiempo total de retardo del controlador y del software: Para la medición del tiempo de retardo del controlador CAN se envía una trama de petición remota y el CCC bajo prueba, que tiene el mensaje de respuesta ya habilitado, procede a la respuesta sin necesidad del software. La Figura 90 muestra el resultado de esta medición para una trama de respuesta con 8 bytes de datos (este tiempo es independiente de la plataforma).

Para la medición del retardo del software, mostrado en la Figura 91, las CPUs reciben una petición de envío, por parte del CCCP, de un objeto de su tabla. Para atender esta petición tienen que acceder a la placa de acceso a bus, leer el mensaje de petición, tomar el mensaje de respuesta correspondiente y ordenar su transmisión. El osciloscopio mide el tiempo que transcurre desde el instante en que envía su trama de petición hasta que se obtiene la trama de respuesta del CCC bajo prueba. Todas la pruebas han sido realizadas con una velocidad de bus de 125 kb/s. En la Tabla 54 se indican los valores obtenidos.

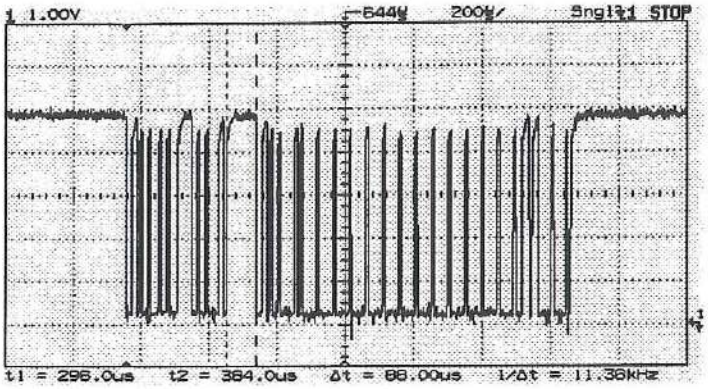
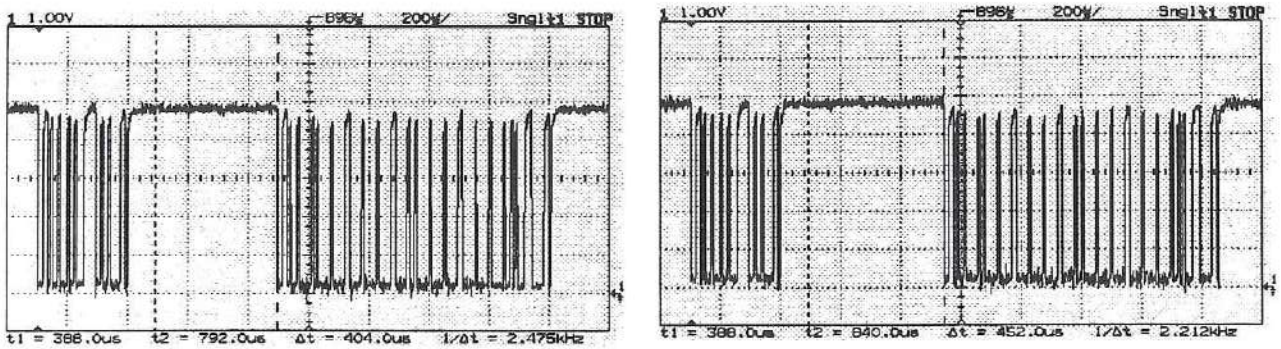


Figura 90: Oscilogramas del tiempo de retardo de la interfaz.



CCC1 CCC2

Figura 91: Oscilogramas del tiempo de retardo del software.

Equipo	Tiempo de retardo de interfaz	Tiempo de retardo de software
CCC1	88 µs	404 µs
CCC2	88 µs	452 µs

Tabla 54: Tiempos de acceso para transmisión.

Medición de la velocidad máxima de funcionamiento: Para encontrar la velocidad máxima de funcionamiento se debe tener en cuenta lo indicado en la Tabla 40. En estas pruebas los dos CCCs reciben un mensaje del CCCP por el objeto de recepción 15 y arrancan un temporizador interno, en el momento de la interrupción de llegada.

Luego la rutina de atención a la interrupción verifica para qué cola de llegada es el mensaje y lo deposita en ella. El temporizador se para cuando el mensaje ha sido depositado en la cola. El tiempo obtenido nos indicará si el CCC bajo prueba tendrá suficiente tiempo para no perder ningún dato si la llegada de los mismos, por el objeto 15, es en ráfaga.

En la Tabla 55 se indican los valores obtenidos.

Equipo	Tiempo de escritura de mensaje en cola
CCC1	218.57 µs
CCC2	235.70 µs

Tabla 55: Tiempos de acceso para recepción.

Para realizar la medida del tiempo en ambas máquinas se utiliza el temporizador número dos del controlador 8254 que incorporan las plataformas PC. Este temporizador nos permite medir intervalos con precisión de 838 ns.

6.2.2 Medición del tiempo máximo de retardo en transmisiones de mensajes no prioritarios.

En estas pruebas suponemos al igual que las simulaciones realizadas en la sección 5.13, que los nodos trabajan en condiciones extremas, es decir, con todas las tablas completas al máximo, pero sin envíos de mensajes con restricciones entre nodos.

Los resultados obtenidos, mostrados en la Tabla 56, indican el peor tiempo de actualización de todos los datos, y se comparan con los resultados calculados en el capítulo 5. Cada una de las pruebas han sido realizadas 100 veces, y en la Tabla 56 se indica el peor de los tiempos obtenidos. En cada una de las 100 pruebas, se ha cambiado, aleatoriamente, el contenido de los datos de los mensajes transmitidos. En la Figura 92 se comparan gráficamente los resultados teóricos y los obtenidos en estas pruebas.

Número total de nodos		**2**	**4**	**8**	**16**
Número total de mensajes		**32**	**96**	**224**	**480**
20 kb/s	**Teórico**	222,7	654,7	1518,7	3246,7
	Medido	**205.4**	**598.3**	**1410.6**	**2935.8**
125 kb/s	**Teórico**	35,6	104,7	243,0	519,4
	Medido	**32.8**	**94.4**	**223.0**	**477.5**

Tabla 56: Tiempos máximos de respuesta en ms.

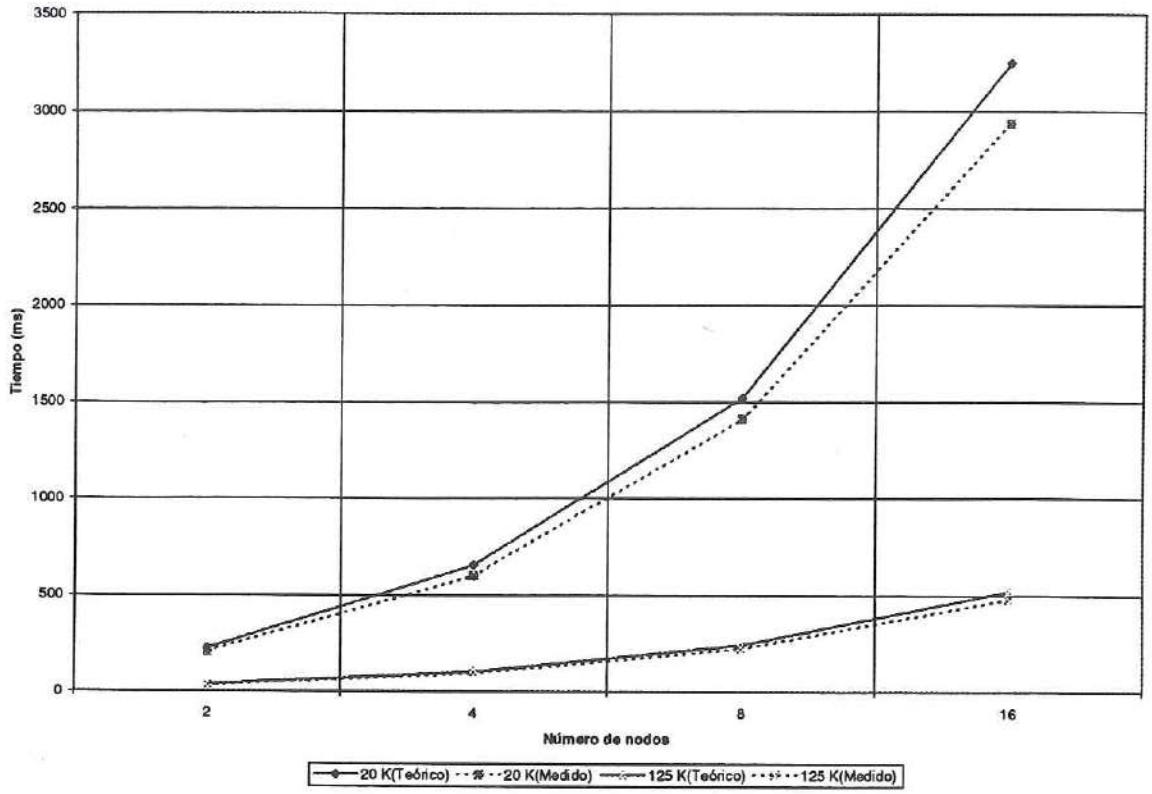


Figura 92: Comparación gráfica de los tiempos máximos de respuestas teóricos y medidos.

6.2.3 Medición del tiempo máximo de retardo en casos de mensajes prioritarios.

Se obtienen los resultados agregando a los mensajes anteriores mensajes con restricciones. Los resultados que muestran el peor tiempo de actualización de todos los datos, se muestran en la Tabla 57. En la Figura 93 se comparan los resultados teóricos con los medidos.

Número total de nodos		**2**	**4**	**8**	**16**
Número total de mensajes		**45**	**121**	**273**	**577**
20 kb/s	**Teórico**	271,5	748,5	1702,5	3610,5
	Medido	**245.3**	**682.6**	**1530.2**	**3263.1**
125 kb/s	**Teórico**	43,4	119,7	272,4	577,6
	Medido	**38.6**	**107.2**	**245.8**	**522.0**

Tabla 57: Tiempos máximos de respuesta en ms.

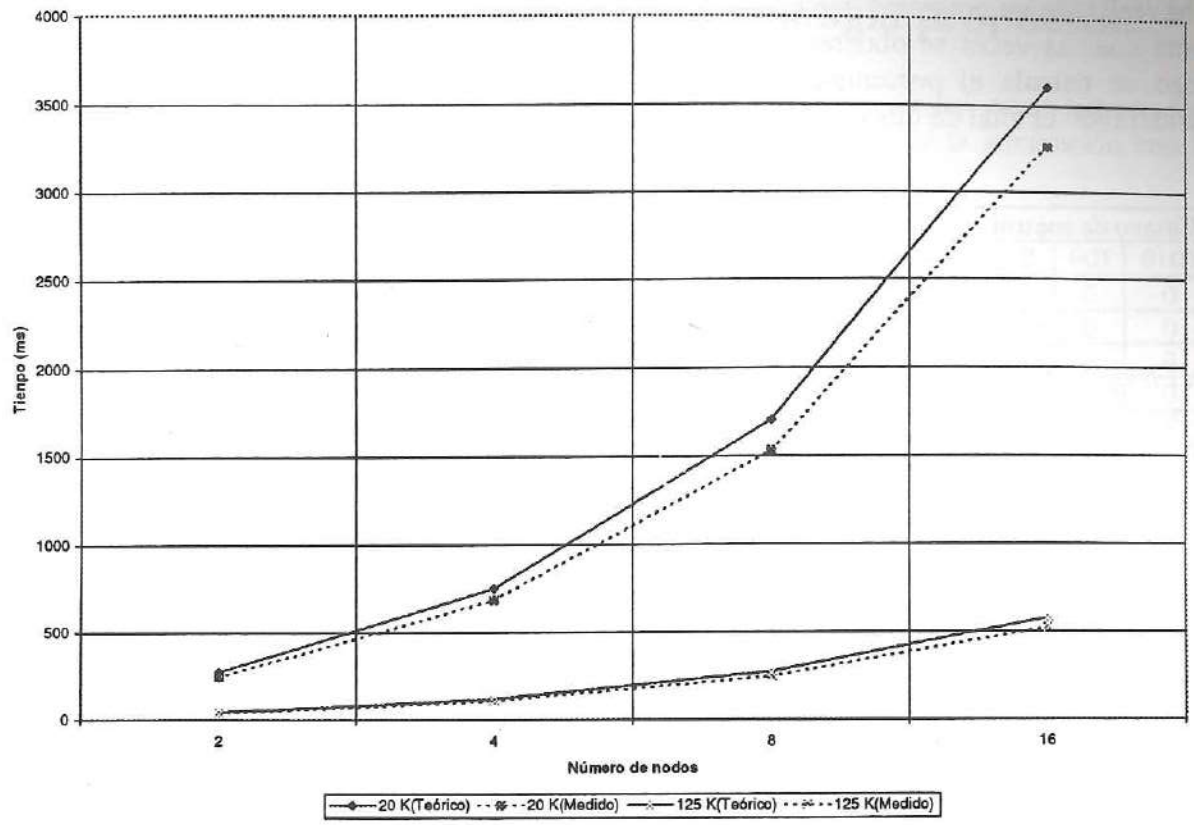


Figura 93: Comparación gráfica de los tiempos máximos de retardo teóricos y medidos.

6.2.4 Análisis estadístico del número de bits de relleno.

Uno de los parámetros que permiten explicar las diferencias obtenidas entre los resultados teóricos y experimentales, es la consideración del número de bits de relleno. En el cálculo teórico analítico se añadió, a cada mensaje, el máximo número de bits de relleno, pero en la práctica ésto es poco probable.

Para tener una referencia de lo que pasa en la realidad hemos realizado un análisis estadístico de los bits de relleno que se pueden encontrar al enviar un objeto CAN en el sistema propuesto.

Para realizar este análisis hemos analizado en primer lugar las combinaciones posibles de identificadores en el sistema. Los resultados se indican en la Tabla 58 donde también se indica (casilleros sombreados) los bits que entrarían en las combinaciones posibles de bits de relleno. Estas 1280 combinaciones son las que existen si tenemos mensajes sin ningún byte de datos.

Al considerar los bytes de datos la cantidad de combinaciones aumenta según se indica en la Tabla 59. Aquí podemos observar que existen 9 casos posibles de mensajes partiendo de uno sin datos a uno con ocho bytes de datos (los números en negritas indican el caso mostrado en la Tabla 58).

Se ha realizado un programa que obtiene 10000 muestras aleatorias de cada uno de los casos y cuenta cuántas veces se obtienen mensajes con un número determinado de bits de relleno. Luego se calcula el porcentaje que estos representan y se hace una estimación media considerando el total de bits de relleno dividido entre el total de muestras.

Campo de control			Campo de identificación y operación.								Número de combinaciones posibles.
ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0	
0	0	0	Reservado								0
0	0	1	Número del mensaje 0 – 255								256
0	1	0	Número del nodo 0 - 15				Tipo de operación				256
0	1	1	Número del nodo 0 - 15				Número de trozo 0 - 15				256
1	0	0	Número del nodo 0 - 15				Tipo de operación				256
1	0	1	Número del nodo 0 - 15				Número de trozo 0 - 15				256
1	1	0	Reservado								0
1	1	1	**No permitido por el estándar.**								
TOTAL			**1280**								

Tabla 58: Número de combinaciones considerando sólo los identificadores (mensaje sin ningún byte de datos).

Número de bytes de datos - combinaciones posibles.								
0	1	2	3	4	5	6	7	8
1280	327680	83886080	2.2×10^{10}	5.5×10^{12}	1.4×10^{15}	3.6×10^{17}	9.2×10^{19}	2.3×10^{22}

Tabla 59: Combinaciones posibles considerando los bytes de datos.

El número medio de bits de relleno, estimado con este procedimiento, se muestra en la Tabla 60, donde también se indica el mismo número calculado en el capítulo 5 de manera teórica.. Hay que indicar que dada la gran cantidad de combinaciones resultantes al añadir el campo de datos, sólo se ha considerado todas las posibilidades reales hasta con mensajes con 1 byte de datos. Para las demás casos hemos generado el valor del dato por segmentos.

Bytes de datos en el objeto	Número de bits de relleno teórico.	Número medio de bits de relleno estimado.
0	8	2.01
1	10	2.12
2	12	2.30
3	14	2.57
4	16	2.60
5	18	2.61
6	20	2.87
7	22	3.28
8	24	3.56

Tabla 60: Número medio de bits de relleno.

Como se puede ver los valores teóricos máximos, que son considerados en el programa de cálculo del tiempo de retardo, distan mucho de los que se podrían encontrar en la práctica, y esta es una de las razones fundamentales para las diferencias de los tiempos de actualización teóricos y los encontrados en este capítulo de verificación.

6.3 Ejemplo de aplicación de la metodología propuesta.

En este apartado se compara la integración realizada, de manera convencional, en un sistema real y en funcionamiento, frente a los resultados que se obtendrían al aplicar la integración con la metodología propuesta.

6.3.1 Red de PLCs en una estación depuradora.

Este es el sistema de supervisión, instalado en la estación depuradora de Barranco Seco en Las Palmas de Gran Canaria, que consta de 11 autómatas de marca SPRECHER. Estos autómatas han sido integrados en un sistema supervisor, mediante el sistema clásico de una red con paso de testigo en anillo tal como se muestra en la Figura 94.

En este sistema se han presentado una serie de problemas, uno de los más comunes es la poca flexibilidad del medio físico, pues los conectores a las tarjetas de interfaz han causado muchos paros de la red. Pero quizás el problema más grave ha sido la reposición de las tarjetas de red, que el fabricante ha discontinuado. Actualmente es posible aún encontrar algunas a precios muy altos. Estos autómatas, al igual que la mayoría de su generación (este autómata lleva instalado 5 años), no cuentan con posibilidad de aceptar tarjetas de red para otro tipo de buses industriales.

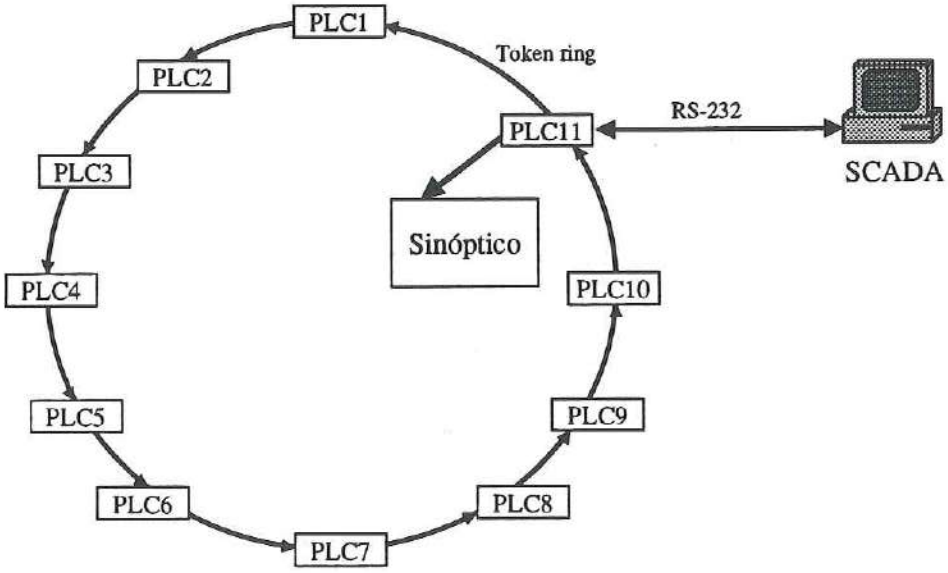


Figura 94: Red de autómatas en la estación depuradora de Barranco Seco.

La solución, con la metodología propuesta, consiste en instalar un módulo CCC en cada autómata formando una red de CCCs, mostrada en la Figura 95, tal como la indicada en el apartado 5.6.

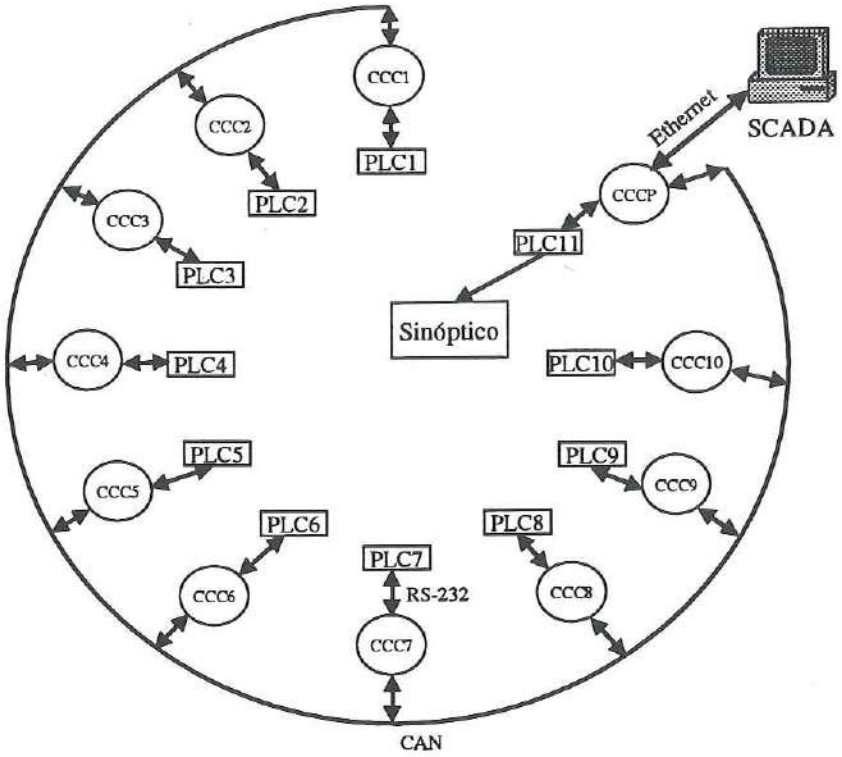


Figura 95: Red de integración propuesta.

6.3.1.1 Descripción de los dispositivos a integrar.

De acuerdo a la disposición física de los nodos se deben definir 11 zonas (indicadas en la Tabla 61. La descripción de cada zona se indica en la Tabla 62.

Zona	PLC	Descripción
1	1	Recirculación de fangos y decantación secundaria
2	2	Espesador de flotación
3	3	Espesador de gravedad
4	4	Calefacción y digestión de fangos
5	5	Cloración y Terciarios
6	6	Turbo Compresores
7	7	Deshidratación
8	8	Recuperación de fangos y decantación primaria
9	9	Bombeo a planta existente
10	10	Recuperación de Energía motores
11	11	Sinóptico (Estación principal)

Tabla 61: Zonas a integrar.

Tipo de dispositivo	Conexión al CCC / (zona)	Protocolo	# Variables a exportar			# Variables a importar			Restricciones de tiempo
			Bit	Byte	Long	Bit	Byte	Long	
PLC SPRECHER	RS-232 / (1)	1	32	0	0	64	0	8	No
PLC SPRECHER	RS-232 / (2)	1	32	0	0	64	0	0	No
PLC SPRECHER	RS-232 / (3)	1	32	0	0	64	0	0	No
PLC SPRECHER	RS-232 / (4)	1	64	0	0	160	0	16	No
PLC SPRECHER	RS-232 / (5)	1	64	0	4	160	0	8	No
PLC SPRECHER	RS-232 / (6)	1	64	0	0	32	0	8	No
PLC SPRECHER	RS-232 / (7)	1	64	0	0	192	0	0	No
PLC SPRECHER	RS-232 / (8)	1	32	0	0	64	0	8	No
PLC SPRECHER	RS-232 / (9)	1	32	0	0	64	0	8	No
PLC SPRECHER	RS-232 / (10)	1	64	0	0	96	0	0	No
PLC SPRECHER	RS-232 / (11)	1	0	0	0	448	0	0	No

Tabla 62: Descripción de los dispositivos a integrar.

6.3.1.2 Entrada de datos en cada uno de los CCCs.

Siguiendo con la metodología en cada CCC, o desde el CCCP, se debe configurar los datos de cada nodo. En la Tabla 63 se muestra un resumen de los referentes a la zona 5 (puesta en negrita en las tablas anteriores).

Nº de referencia de la variable	Descripción	Tipo de variable	Driver	Puerto	Envío / recepción	Nodo(s) de destino Nodo de origen	Restric-ciones
1	Ext. cloro 1	b	1	1	E	P	No
2	Ext. cloro 2	b	1	1	E	P	No
3	Bomba cloro 1	b	1	1	E	P	No
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
234	Clorador 1	L	1	1	R	P	No
235	Clorador 2	L	1	1	R	P	No
236	Presión agua	L	1	1	R	P	No

Tabla 63: Entrada de datos en CCC de zona 5.

6.3.1.3 Prestaciones con el método propuesto.

Con la metodología propuesta en esta tesis se debería añadir a cada autómata un CCC que implemente el protocolo de este tipo de autómatas. La Tabla 64 indica el número total de objetos que cada CCC intercambiará con el CCCP en el caso en que se requiera exportar al SCADA todas las variables del sistema (entre paréntesis se indica el número de bytes del objeto). El agrupamiento de variables es realizado según se especificó en 5.9.2. En esta

aplicación el nodo CCCP debe estar en el autómata 11 que es el que está físicamente más cerca del sistema SCADA.

CCC	# Objetos a exportar por variables tipo:			# Objetos a importar por variables tipo:		
	Bit	Byte	Long	Bit	Byte	Long
1	1(4)	0	0	1(8)	0	2(8)
2	1(4)	0	0	1(8)	0	0
3	1(4)	0	0	1(8)	0	0
4	1(8)	0	0	2(8), 1(4)	0	4(8)
5	1(8)	0	1(8)	2(8), 1(4)	0	2(8)
6	1(8)	0	0	1(4)	0	2(8)
7	1(8)	0	0	3(8)	0	0
8	1(4)	0	0	1(8)	0	1(8)
9	1(4)	0	0	1(8)	0	2(8)
10	1(8)	0	0	1(8), 1(4)	0	0
Total	5(8), 5(4)	0	1(8)	13(8), 4(4)	0	13(8)

Tabla 64: Objetos a intercambiar con el CCCP.

Con el programa PCTRM descrito en el capítulo 5, y el total de objetos indicados en la Tabla 64, obtenemos que en las peores condiciones la tabla en el CCCP se actualizará a 125 kb/s cada 41,68 ms, y a 20 kb/s cada 260,5 ms. Estos tiempos obtenidos están garantizados para todas las condiciones de funcionamiento del sistema. También con este tipo de solución no existen problemas de flexibilidad del medio, pues el par de cables trenzados utilizados en la red de CCCs es de los medios más flexibles para este tipo de entorno.

6.3.2 Conclusiones.

Si bien las mediciones realizadas en este capítulo no abarcan todos los casos posibles, constituyen una demostración de la utilidad del sistema propuesto. Además el ejemplo de aplicación es representativo de los sistemas que normalmente se integrarán con este tipo de solución propuesta.

Para el ejemplo de aplicación podemos indicar, como ventaja, que este tipo de integración puede ser realizada a la quinta parte del costo de la solución actual.

También se puede decir, que cuando el sistema lo permita, el CCCP se debe conectar en la zona que debe transferir la mayor cantidad de datos al SCADA, pues de esta manera se evitará el tráfico relativo a estas variables en la red de CCCs.

Capítulo 7.

Conclusiones y líneas de investigación futuras.

7.1 Conclusiones.

En la presente tesis se propone una metodología para la integración de sistemas heterogéneos de comunicaciones de campo, para los casos en los cuales el tamaño del sistema no justifique el uso de un bus de campo de elevadas prestaciones y donde los sistemas de integración convencionales no cubran todos los requerimientos de la aplicación. Para realizar esta propuesta en primer lugar se han realizado los siguientes estudios:

- Estudio sistemático de los requisitos y prestaciones de los sistemas de comunicación industriales, donde se han analizado los medios de transmisión, estándares de comunicación, topologías de redes, protocolos de control de acceso y modelos de operación de redes.

- Estudio y análisis de las redes de área local utilizadas en entornos automatizados incluyendo las redes de área local de propósito general utilizadas en este tipo de entornos y las redes de área local industriales con especial énfasis en WorldFIP y CAN.

Como resultado de los estudios antes mencionados proponemos una metodología sistemática y flexible, para la integración de comunicaciones de campo, consistente en concentrar las variables de los elementos a integrar en un dispositivo inteligente que las agrupe y transfiera al sistema integrador según las restricciones de las mismas.

- Para posibilitar la implantación de la metodología se ha especificado un dispositivo, llamado Concentrador de Comunicación de Campo CCC, que, por una parte se comunica con los elementos a integrar, y por otra puede presentar al sistema de integración la información requerida en forma de una tabla que refleja todo lo que sucede en el sistema integrado, permitiendo la comunicación bidireccional.

- Esta metodología – al igual que el enfoque "*top down*" integrador por vía de estandarización- requiere una completa planificación "a priori" de los sistemas, pero en cambio utiliza un enfoque "*bottom up*", integrador por vía de interfaz, de mayor flexibilidad y que resuelve también la integración "a posteriori" de sistemas.

Las comunicaciones entre CCCs se estructuran mediante una red de campo que utiliza el protocolo CAN estandarizado en los niveles OSI 1 y 2 según ISO11519 e ISO11898 pues como se demuestra en el capítulo 5 es el que más se adecua a los requerimientos de la propuesta. Esta red se caracteriza por las siguientes prestaciones:

1. Permite la comunicación bidireccional y transparente entre todos los dispositivos de campo que así lo necesiten, sin importar la red ni el protocolo que soporten.

2. Garantiza los tiempos, y las restricciones de consistencia temporal, en los mensajes críticos entre nodos CCC, cuando se necesiten transferencias en tiempos máximos definidos.

3. Permite el intercambio de datos bidireccionales con los otros niveles del sistema de comunicación.

4. Permite la configuración remota de los nodos, desde el que será el CCC principal, o bien desde los niveles superiores.

5. Está basada en la utilización de CCC de arquitectura compacta, modular y de bajo coste para integrar y facilitar las comunicaciones en entornos industriales.

Se ha desarrollado el software necesario para permitir la transferencia de las tablas de objetos de comunicación locales de todos los CCC a uno principal CCCP, que es el encargado de las transferencias al nivel superior ya que los sistemas comerciales que utilizan CAN como SDS DeviceNet, etc. no están optimizados para realizar transferencias como las que se proponen en la metodología.

Esta tesis aporta alternativas eficaces de integración en los siguientes casos:

1. En todos aquellos sistemas que ya cuentan con elementos heterogéneos distribuidos a lo largo de toda la planta y de los que se requiere su integración para su control o supervisión. Un ejemplo son los sistemas que han crecido por aumento de producción (caso de líneas de fabricación construidas en fases sucesivas).

2. En los sistemas que están en la fase de diseño pero que, debido a las imposiciones de los fabricantes de equipamiento industrial, resulta imposible adoptar un único estándar de comunicación. Esto ocurre comúnmente porque en los equipamientos industriales normalmente se implementa un sistema de comunicación que se adecua a los requerimientos del fabricante y no a los del usuario.

3. En aplicaciones que dispongan de *drivers* de comunicación con los dispositivos, pero donde las distancias físicas de separación, de éstos al elemento de integración, impiden el uso de las interfaces estándares de los mismos. Normalmente el elemento integrador será un PC, cuyas interfaces estándares - generalmente RS232 -, no son capaces de alcanzar las distancias usuales dentro de los entornos automatizados.

4. En las aplicaciones donde algunas variables requieren ser transferidas cumpliendo restricciones, por ejemplo de tiempo o de consistencia, que los sistemas de integración usuales no garantizan.

5. En aquellos casos en que los elementos a integrar soportan una interfaz propia de comunicación, por ejemplo autómatas que tienen capacidad de utilizar interfaces para buses de campo - como por ejemplo FIP, u otros protocolos -, pero donde la cantidad y flujo de datos que se desea exportar no justifican el uso de los mismos; esta justificación puede ser tanto en costo por nodo como por eficiencia del bus.

6. En los casos en que la fabricación de los elementos de comunicación ha sido discontinuada por los fabricantes, o el mantenimiento resulta costoso por el tipo de hardware utilizado.

7. Al integrar equipos o procesos aislados, cuyas especificaciones iniciales no contemplaban el trabajar conjuntamente con otros elementos dentro del entorno.

8. En cualquier combinación mezcla de las anteriores, cuya integración siempre será posible con la metodología propuesta en esta tesis.

Para demostrar la viabilidad de la metodología se han desarrollado dos versiones de CCC, una sobre arquitectura PC y otra sobre arquitectura Motorola 68000, que incluyen sus correspondientes drivers. Con estos CCCs se han realizado pruebas de laboratorio obteniendo las siguientes conclusiones:

1. En las pruebas realizadas a 20 kb/s (según el estándar se puede alcanzar 3.3 km) se observa que el tiempo de actualización de un sistema con 16 nodos es de menos de 3 segundos y medio, tiempo adecuado para la mayoría de aplicaciones.

2. El sistema también permite realizar transferencias rápidas. Como se demuestra en el apartado 5.13.4 se pueden enviar variables cíclicas con periodos muy breves de repetición, lo que garantiza la viabilidad del sistema propuesto en aplicaciones exigentes.

3. También de las pruebas realizadas se observa que la variable que más influye en el tiempo de actualización es la cantidad de mensajes, más que la longitud de los mismos.

4. La velocidad máxima que se puede alcanzar en el bus está limitada por la velocidad de la CPU que gestiona el chip de bus CAN. Para liberarla de la gestión de las interrupciones por mensajes, en los casos de sistemas con carga excesiva se recomienda el uso de un CCCP con interfaz a microcontrolador que ya tenga integrado el chip de interfaz al bus.

5. El sistema facilita conectar y desconectar nodos, aun bajo tensión, pudiendo ser implantado con pocas o ninguna parada de los elementos industriales interconectados.

6. Se ha comprobado que se puede planificar cualquier situación de integración real que se pueda presentar en un sistema automatizado de mediana envergadura. Este es el caso más usual, pues son pocas las industrias que alcanzan el nivel de datos y transferencias, para los cuales se hace necesaria la implantación de un sistema de comunicación según el modelo completo de CIM.

7.2 Líneas de investigación futuras.

Actualmente en el laboratorio de Equipos y Productos Industriales del CMA se está trabajando en el diseño de tarjetas inteligentes con bus CAN tanto para PC como para tarjetas con CPU propia.

Se está ampliando la funcionalidad del sistema de autoconfiguración y configuración remota del nodo CCCP para hacer el funcionamiento más flexible. Igualmente se está añadiendo al software de transferencia de datos mecanismos de envío de reconocimiento de mensajes entre nodos.

Para aplicaciones con requerimientos mínimos, por ejemplo exportar algunas variables de un autómata, se están realizando los estudios para implementaciones de redes de CCCs muy simples – circuito de interfaz a bus más microcontrolador pequeño -. Éstos solamente tienen un proceso de atención a los dos puertos (el del CCCP y el del dispositivo a integrar) que transfiere los datos de un puerto a otro sin ninguna clase de tratamiento. El driver correspondiente está instalado en el CCCP estableciendo canales serie virtuales de comunicación entre éste y los CCCs.

La implantación física de la metodología propuesta en el sistema de supervisión de los 11 autómatas de la estación depuradora de Barranco Seco (ubicada en Las Palmas de Gran Canaria) permitirá disponer de una planta de experimentación real a gran escala que será fuente de sugerencias, innovaciones y mejoras del sistema.

También está en desarrollo la tesis "Integración de sistemas y medios de comunicación en entornos industriales de supervisión y control distribuido" complementaria a la aquí presentada.

El CMA de la ULPGC forma parte del grupo temático "COMUNICACIONES PARA EL CONTROL DISTRIBUIDO", formado a nivel nacional por centros de investigación que realizan trabajos en el área. Se pretende continuar y fomentar la cooperación en el seno del grupo para la realización de trabajos similares con buses de campo de mayores prestaciones, como por ejemplo, FIP y PROFIBUS.

Por ultimo los trabajos experimentales desarrollados en esta tesis permiten fácilmente definir módulos y software didácticos para difundir el uso de los mecanismos de integración con bus CAN.

Glosario de términos.

ANSI	*American National Standard Institute*
AP	*Application Process*
ARCnet	*Attached Resource Computer Network*
ASCII	*American Standard Code for Information Interchange*
ASI	*Actuator Sensor Interface*
ATM	*Asychronous Transfer Mode*
BRD	*Broadcast & Receive Device*
CAN	*Control Area Network*
CCE	*Cime Computing Environment*
CIM	*Computer Integrated Manufacturing*
CNMA	*Communications Network for Manufacturing Applications*
CPU	*Central Processing Unit*
CRC	*Cyclic Redundancy Check*
CSMA	*Carrier Sense Multiple Access*
DDE	*Dynamic Data Exchange*
DDL	*Data Definition Language*
DDLM	*Direct Data Link Mapper*
DLC	*Data Length Code*
DML	*Data Manipulation Language*
EDF	*End of Frame*
EIA	*Electronics Industries Association*
EMI	*Electromagnetics Interferences*
FDDI	*Fiber Distributed Data Interface*
FDL	*Fieldbus Data Link*
FEP	*Front End Processor*
FIP	*Flux Information Processus*
FMS	*Fieldbus Message Specification*
GPIB	*General Purpose Intrument Bus*
HART	*Highway Addressable Remote Transducer*
IDE	*Identifier Extension*
IEC	*International Electrotechnical Commission*
IEEE	*Institute of Electrical and Electronics Engineers*
IFS	*Interframe Space*
IMMI	*Intelligent Man Machine Interface*
ISA	*Instrument Society of America*
ISO	*International Standard Organisation*
LAN	*Local Area Network*
LLI	*Low Layer Interface*
LONwork	*Local Operating Network*

MAP	*Manufacturing Automation Protocol*
MMS	*Manufacturing Messagge Specification*
OSI	*Open System Interconnection*
PC	*Personal Computer*
PDU	*Protocol Data Unit*
PID	*Proportional-Integral-Differential control*
PLC	*Programmable Logic Controller*
P-NET	*Process automation NET*
PROFIBUS	*Process Field Bus*
RAM	*Random Access Memory*
RCSMA	*Reservation CSMA*
ROM	*Read Only Memory*
RTR	*Remote Transmission Request*
RTU	*Remote Telemetry Unit / Remote Terminal Unit*
SCADA	*Supervision Control And Data Acquisition*
SLIO	*Serial Linked I/O*
SNA	*System Network Architecture*
SOF	*Star Of Frame*
SQL	*Structured Query Language*
STP	*Shielded Twisted Pair*
TCP/IP	*Transmission Control Protocol / Internet Protocol*
TDMA	*Time Division Multiple Access*
UHF	*Ultra High Frecuencia*
USB	*Universal Serial Bus*
UTP	*Unshielded Twisted Pair*

Referencias.

[Alco92] X. Alcober, "Las comunicaciones MAP y los protocolos MMS", Automática e Instrumentación, Noviembre 1992, No 228, pp. 80-85.

[ANFD80] ANSI X3T9.5 1980, "Fiber Distributed Data Interface FDDI".

[ArRa91] K. Arvid, K. Ramamrithan, "A local area network architecture for communication in distributed real-time systems", Journal of Real-Time Systems, Vol. 3, No. 2, May 1991, 238-243.

[AuBu93] N. Audsley, A. Burns, M. Richardson, "Applying New Scheduling Theory to Static Priority Pre-emptive Schedulling", Software Engineering Journal 8, September 1993, pp. 284-292.

[Axel95] J. Axelson, "Networks for Monitoring and Control Using an RS-485 interface", Microcomputer Journal, July/August 95, pp. 27-36.

[Azev96] J. Azevedo, "WorldFIP: Protocol update", version 1, document No. WF/IRW/003/02, France 1996.

[BeSi92] Y. Benkhellat, M. Siebert, "Interoperatibility of sensors and distributed systems", Eurosensor 92 , San Sebastian SPAIN, 1992, pp. 165-169.

[Bhar94] P. Bhargav, "Communication Protocols for Embedded Systems", Embedded Systems Programming No. 7, November 1994, pp. 46-58.

[Bowd96] R. Bowden, "HART a Technical Description", Fisher-Rosemount, second edition, Germany, 1996.

[BrDa91] J. Bron, K. Daoudi, and M. Veron, "Temporal Performance Evaluation of the MMS Services Supported by MAP Network", William G. Sullivan Editors, 1991, pp. 302-312.

[BuNi93] A. Burns, M. Nicholson, K. Tindell, "Allocating and Scheduling Hard Real-Time Task on a Point-to-Point Distributed System", Proceedings of the Workshop on Parallel and Distributed Real-Time Systems, California, April 1993, pp. 11-20.

[BuTi94] A. Burns, K. Tindell, "Fixed Priority Scheduling with Deadlines Prior to Completion", Proceedings Sixth Euromicro Workshop on Real Time Systems, IEEE Computer Society Press, Sweden, June 1994, pp. 138-142.

[CAN2.0] Robert Bosch GmbH, "CAN: specification", ver.2.0, Stuttgart, 1991.

[CaSi95] C. Cardeira, F. Simonot, M. Bayart, "Intelligent Field devices and Field Buses: Impact on Applications Design Methodology", Proceddings ASI'95, Cascals, Portugal, June 1995, pp 241-246.

[CaSt96] S. Cavalieri, A. Di Stefano, L. Lo Bello, O. Mirabella, "CAN Assessment in Time-Critical Cyclic Applications through Petri Net Model", IEEE Computer and Communications, 1996, No. 6, pp. 922-927.

[Chen89] L. Chen, "Reservation CSMA/CD: A Multiple Access Protocol for LANs", IEEE Journal on Selected Areas in Communications, February 1989, pp. 135-142.

[Deco85] J. Decotignie, "Hierarchy of Industrial Local Area Networks: towars a sensor and actuator network"; Journes d'Electronique 85, Lausanne Oct. 1985, pp. 305-314.

[DePl87] J. Decotignie, P. Pleinevaux, "Field Bus in the Hierarchy of Factory Communications: the Limits of a Classical Approach"; Workshop on Factory Communications, N.B.S., Maryland March 1987, pp. 3-16.

[DePl93] J. Decotignie, P. Pleinevaux, "A Survey on Industrial Communications Networks"; Annales des telecommunications, invited paper, Vol. 48, No. 9-10, 1993, pp. 435-448.

[DIN19245] PROFIBUS standars DIN 19245 part 1,2,3,4.

[Eche95] Echelon Corporation, "Determinism in Industrial Computer Control Network Applications", January 1995.

[EIA69] EIA Standard RS-232-C (August 1969), "Interface Between Data Terminal Equipment and Data Communication Equipment Employing Serial Binary Data Interchange".

[EIAB75] EIA Standard RS-422 (April 1975), "Electrical Characteristics of Balanced Voltaje Digital Interface Circuits".

[EIAU75] EIA Standard RS-423 (April 1975), " Electrical Characteristics of Unbalanced Voltaje Digital Interface Circuits".

[EMUG89] European MAP Users Group. "Users Requeriments for Communications in Time Critical Applications", EMUG Technical Note, February 1989.

[FuAl92] J. Fuertes, X. Alcober, "Comunicaciones Industriales", Automática e Intrumentación, Diciembre 1992, No 228, pp.79-85.

[GeLe94] M. Gerla, A. Lessard, "Wireless communication in the automated factory enviroment", IEEE Network, No 3, 1994, pp. 64-69.

[Gene88] General Motors, "Manufacturing Automation Protocol V3.0", GM, 1988.

[Gillu94] F. Gilluwe, "The UNDOCUMENTED PC", Addison-Wesley Publishing Company, 1994, ISBN 0-201-62277-7.

[HePa93] J. Hennessy, D. Patterson, "Arquitectura de computadores: Un enfoque cuantitativo", Mc. Graw Hill 1993, ISBN: 84-7615-912-9.

[HoTh88] K. Hoswell, G. Thomas, "ARCnet Factory LAN premier", Second Printing, Downers Grove, Illinois, 1988.

[HuLá89] A. Huertas, C. Lázaro, "Buses de campo: Estado de la tecnología", Automática e Intrumentación, Enero 1989, No 187, pp. 141-155.

[IEEECS85] ANSI/IEEE Std. 802.3-1985, "Carrier Sense Multiple Access with Collision Detection (CSMA/CD): Access Method and Physical Layer Specifications".

[IEEEL85] ANSI/IEEE Std. 802.2-1985, "Carrier Sense Multiple Access with Collision Detection (CSMA/CD): Logical Link Control".

[IEEETP85] ANSI/IEEE Std. 802.4-1985, "Token Bus Access Method and Physical Layer Specifications".

[IEEETR85] ANSI/IEEE Std. 802.5-1985, "Token Ring Access Method and Physical Layer Specifications".

[IN-82517] INTEL, "82517 Serial Communications Controller: Architectural Overview", Intel Corporation, January 1996.

[ISA359E] ISA-1990-359E, Industrial automation systems – Systems integration and communications – Fieldbus, Part V: Data link service specification.

[ISA360C] ISA-1990-360C, Industrial automation systems – Systems integration and communications – Fieldbus, Part VI: Data link protocol specification.

[ISO8650] ISO/IEC 8650: 1987, Information processing systems – Open systems interconnection – Protocol specification for the Association Control Service Element.

[ISO9072-2] Information processing systems – Text communications – Remote operations, Part2: Protocol specification.

[ISO9506-1] 1990, Industrial automation systems – Systems integration and communications – (MMS) Manufacturing messaging specification, Part1: Service definition.

[ISO9506-2] 1990, Industrial automation systems – Systems integration and communications – (MMS) Manufacturing messaging specification, Part2:

Protocol specification.

[ISO9596] ISO/IEC 9596: 1990, Information technology – Open sistems interconnection – Common management information protocol (CMIP) specification.

[LaCr90] P. Lagoni, C. Crall, T. Bartz, "HP MAP 3.0 Manufacturing Message Specification/800", Hewlett-Packard Journal, August 1990, pp. 31-39.

[LeWh82] J. Leung, J. Whitehead, "On The Complexity of Fixed-Priority Scheduling of Periodic Real-Time Task", Performance Evaluation (Vol 2), Part 4, December 1982, pp. 237-250.

[LoMa95] P. Lorenz, Z. Mammeri, "Real-Time Software Architecture: Application to FIP Fieldbus", Procedings 3rd. IFAC/IFIP Workshop on Algoritms and Architectures for Real-Time Control, OSTEND (Belgium), 1995, pp. 415-423.

[LON91] LonWorks Engineering Bulletin, "Enhanced Media Acces Control", Echelon Corp. August 1991.

[MAP3.0] Society of Automotive Engineer, "Manufacturing Automation Protocol, MAP 3.0: Specification", 1988.

[MoMu95] P. Morel, R. Muralt, "Wireless Extension for Fielbus", 2nd Int. Conf. on Industrial Automation, Nancy – France, June 1995, pp 10-14.

[OLCHFA] "The OLCHFA Project: Industrial Needs For Time-Critical Wireless Communications" http://www.faps.uni-erlangen.de:1200/persons/solvie /solvie_olchfa-e.html.

[OSI83] The OSI Reference Model, Proceedings of the IEEE, Vol. 71, No. 12, December 1983.

[PaLe94] H. Park, C. Lee, W. Kwon, "Analysis on the User's Response Time for Mini-MAP Systems", Proc. Of IFAC Worshop on DCCS, Toledo, Spain, Sept. 1994, pp. 35-41.

[PhBr91] T. Phinney, P. Brett, D. McGowan, "Fieldbus – Real-Time Comes to OSI", IEEE network Vol 5, 1991, pp.594-599.

[Pint94] J. Pinto, "Fielbus: A Neutral Instrumentation Vendor's Perpective", ISA proceding 1994, http://www.actionio.com/jimpinto/fbarticl.html.

[P-NET] "Danish National Pre Standard: Specification", published by the International P-NET User Organization.

[Rome90] J. Romeral, "Bus de campo para la interconexión del proceso con sistemas digitales de control", Automática e Instrumentación, Enero 1990, No 198, pp.

141-147.

[RuDe94] L. Ruiz, J. Decotignie, "Fieldbus: a Network for Process Level Communication", IEEE Industrial Electronics Society Newsletter, Vol. 41, No 3, September 1994, pp. 4-6.

[SaMa95] G. Saba, Z. Mammeri, "Some Solutions for FIP Network Interconnection", Proc. 1st. IEEE Workshop on Factory Communications Systems, Lausanne, 1995, pp.13-20.

[SaTh93] G. Saba, J. Thomesse, Y. Song, "Space and Time Consistency Qualifications in a Distributed Communication System", Proceedings of IMACS/IFAC second International Symposium on System Simulation, Brussel, Vol 1, April 1993, pp. 383-391.

[SiSo93] F. Simonot, Y. Song, J. Thomesse, "On message sojourn time for TDM schemes with any buffer capacity", IEEE Transactions on Communications 43, April 1995, pp.1013-1021.

[SiTh94] M. Siebert, J. Thomesse, "Interworking of Fielddevices", Proc. IFAC Symposium on Intelligent Components and Instruments for Control Application, Budapest, June 1994, pp. 98-103.

[SoMa94] Y. Song, J. Martyr, R. Schott, "Adding local scheduling mechanisms to FDDI for time critical communications", Proceedings of IECON'94, Bologna, Sept. 1994, pp. 1190-1195.

[SoTh94] Y. Song, J. Thomesse, "Classification of Networks", CEN Workshop on Industrial Communications, Paris, Septembre 1994, pp. 51-58.

[Stal91] W. Stallings, "Data and Computer Communications", 3rd. ed., Mcacmillan, New York, 1991.

[Tane91] A. Tanenbaum, "Redes de Ordenadores", 2nd. Edición., Prentice Hall, Mexico 1991, ISBN: 0-13-162959-X.

[TCCA92] ISO/TC184/SC5/WG2/T.C.C.A. Technical report of the TCCA rapporteurs' group identifying user requirements for systems supporting time-critical communications., Doc. No. 73, 1992.

[Thom97] L. Thompson, "Industrial Data Communications", Instrument Society of America, 2nd edition, 1997, ISBN: 1-55617-585-X.

[TiBu229] K. Tindell, A. Burns, "Guaranteed Message Latencies for Distributed Safety Critical Hard Real-Time Networks", Technical report YCS229, Department of computer Science, University of York, England, May 1994, 17 pages.

[TiBu94] K. Tindell, A. Burns, "Guaranteeing Message Latencies on Control Area

Network", Procedints 1st International CAN Conference, Mainz Germany, September 1994, pp.21-30.

[TiCl94] K. Tindell, J. Clark, "Holistic Schedulability Analysis for Distributed Hard Real Time Systems", Microprocessors and Microprogramming Nº 40, 1994, pp. 117-134.

[Tind93] K. Tindell, "Fixed Priority Scheduling for Hard Real Time Systems", Dphil Thesis, YCST 94/03, Department of Computer Science, University of York (1993).

[UlDe88] G. Ulloa, J. Decotignie, "Field Bus: a real alternative for time-critical networks", Proceedings EFOC/LAN 88, Ansterdand, June 1988, pp. 334-337.

[UpDe96] P. Upender, A. Dean, "Variability of CAN network Perfomance", Proceedings of the 3rd. International CAN conference, Paris – France, October 1996, pp. 16-23.

[USB96] INTEL, "Universal Serial Bus (USB): Specification", Revision 1.0, January 15, 1996.

[VaCi90] A. Valenzano, L. Ciminiera, "Performance evaluation of MiniMAP networks", IEEE trans. On Industrial Electronics, Vol. 37, No 3, June1990, pp. 253-258.

[VaDe92] A. Valenzano, C. Demartini, L. Ciminiera, "MAP and TOP Communications", Addison Wesley Publishing Company, 1992.

[ViBe94] K. Vijayananda, G. Berneth, P. Raja, "An Architecture of a MAP Stack for Real-Time Programming", Proceedings 19th IFAC/IFIP Workshop on Real-Time Programming, WRTP-94, Isle of Reichnau, Germany, June 1994, pp. 1-7.

[WorldFIP] WorldFIP: French National Fieldbus Specification NFC 46-602, -603, -604, -605, -606, -607.