Tema 1 Introducción a los sistemas en tiempo real

1.1 Introducción

De todos es conocida la fuerte presencia de los sistemas basados en microprocesadores, especialmente las computadoras, en campos muy diversos como la automatización industrial, el cálculo científico, la gestión de información, el ocio y otras servicios usuales de la vida cotidiana. En la mayor parte de las ocasiones la imagen que se tiene de estos sistemas es un computador, con monitor y resto de periféricos, atendido por algún operador para solicitarle la realización de la tarea deseada.

Menos evidente es el uso de las computadoras como parte funcional de un sistema electromecánico más complejo y que funcione con una cierta autonomía, entendida en el sentido de no precisar constantemente de la presencia de un operador. En un computador la unidad de proceso, el microprocesador, se ha incrustado junto con otros elementos periféricos (disco duro, monitor, teclado, tarjeta de vídeo, tarjeta de sonido, puertos serie y paralelo etc.) para formar un todo que sería la imagen típica de un ordenador; de igual manera podemos encontrar sistemas donde un micro, más o menos complejo, forma un todo con el sistema con el que se relaciona. Además lo usual es que estén físicamente ocultos y su acción sea transparente[1] al usuario. Un ejemplo muy sencillo podría ser el micro que controla la secuencia de programas de una lavadora, u otros mucho más complejos como los que gobiernan robots, vehículos autónomos, aviones y naves espaciales o sistemas de seguimiento automático por radar.

La naturaleza de estos sistemas incrustados (*embedded* en la terminología inglesa) es la que nos lleva al tiempo real (*real-time*). El computador incrustado está normalmente ejerciendo el control sobre todo el sistema que está en relación con el "mundo real". La capacidad de entender y reaccionar ante eventos reales a medida que se producen es esencial para un funcionamiento correcto. Los sistemas que deben responder ante los eventos reales que se producen en su entorno se conocen como sistemas reactivos.

Los términos tiempo real e incrustado ofrecen un alcance amplio y no perfectamente definido por todos los autores. Aquí no haremos distinciones y comenzaremos por comprender qué caracteriza a estos sistemas antes de poder pasar a ver cómo se pueden diseñar y desarrollar.

Habitualmente los sistemas en tiempo real caen en la categoría de sistemas dedicados. Quiere esto decir que han sido diseñados pensando en una aplicación específica y previendo un tiempo de vida determinado. Cuando alguien solicita la concepción y desarrollo de un sistema en tiempo real presenta unas especificaciones para que el sistema realice una serie de tareas muy concretas. No se tienen en cuenta sistemas generales como podría ser un ordenador personal, preparado para ejecutar software muy diverso a cualquier hora del día. En los sistemas que nos ocupa la relación del microprocesador con el entorno en el que se encuentra hace que los desarrollos se deban realizar a medida para cada aplicación.

La mayor parte de los autores no se ponen de acuerdo en una definición precisa de lo que sería un sistema en tiempo real, marcando incluso diferencias entre sistemas en tiempo real y sistemas incrustados, mientras que otros autores hacen referencia a ellos como "real time and embedded" o abreviadamente RT&E [1]. A partir de ahora no realizaremos tal distinción y nos referiremos abreviadamente a este tipo de sistemas con el nombre genérico de sistemas en tiempo real (STR).

La definición de sistema en tiempo real ofrecida por el Diccionario Oxford de Computación es: 'Cualquier sistema en el cual el tiempo en que se produce la salida es significativo, debido usualmente a que la entrada se corresponde con alguna variación dentro del mundo físico, y la salida está relacionada con tal variación. El retardo entre el tiempo de la entrada y el de la salida producida debe ser pequeño para un funcionamiento adecuado'.

Esta definición cubre una amplia variedad de sistemas; por ejemplo, desde estaciones de trabajo Unix, en las que el usuario espera obtener respuesta en unos segundos, hasta sistemas de control de vuelo, en los cuales un retraso en la respuesta puede suponer un desastre.

Cooling[4] ofrece la siguiente definición: “Sistemas de tiempo real son aquellos que deben producir respuestas correctas dentro de un intervalo de tiempo definido. Si el tiempo de respuesta excede ese límite, se produce una degradación del funcionamiento y/o un funcionamiento erróneo”.

[1] Se dice que un elemento o acción es transparente al usuario cuando éste no tiene control sobre la acción de ese elemento

1.2 Los STR y sus características

Antes de comenzar a diseñar y desarrollar STR es necesario establecer lo que se entiende por sistema y cuales son las características especiales que lo convierten en un STR.

Diremos que un sistema es una entidad coherente [1], cuyos constituyentes están combinados para formar algo, que no solamente es mayor que los constituyentes individuales, sino que sus propiedades se extienden más allá que las de los elementos individuales que componen el sistema. Una forma práctica de adquirir un conocimiento de la naturaleza de los STR consiste en establecer una forma de clasificación que incluya todos los elementos de más interés (y posiblemente algunos otros), atendiendo a sus características y propiedades más relevantes. El objetivo al definir una taxonomía es establecer un conjunto de reglas que permitan clasificar e identificar STR. Una vez elegidas las características para la clasificación, se deberá establecer el rango de valores aceptado para cada una de las variables que representan al sistema dentro de cada característica escogida. Por supuesto es complicado decidir qué propiedades estudiar y establecer un rango de variación dentro de ellas. En el trabajo COMPLEMENT [1] se adoptó establecer 5 características primarias, 6 características secundarias y 5 características concretas. Dado el alto número de características se decidió, para no complicarlo aún más, que los valores que tomaran las variables fueran solamente alto y bajo, de esta forma se obtienen 2^N categorías siendo N el número de características escogidas.

En primer lugar analizaremos las características primarias, estableciendo una clasificación con un sistema representativo para cada una de las categorías. Si se pretende establecer clasificaciones más finas se puede estudiar el comportamiento de los sistemas atendiendo a las características secundarias y concretas.

1.2.1 Características primarias

Las 5 características adoptadas, junto con los valores posibles que pueden tomar, se ven en la siguiente tabla.

Característica descriptiva	Conjunto de valores	
Procesamiento concurrente	Muchos	eventos concurrentes
	Pocos	eventos concurrentes
Interfaz Hardware	Significativa	interfaz con el hardware
	Insignificante	interfaz con el hardware
Tiempo de reacción ante los eventos	Rígido (hard)	el sistema se considera que ha fallado si responde demasiado tarde ante los eventos externos
	Flexible (soft)	responder tarde ante un evento es indeseable pero no se considera un fallo
Arquitectura distribuida	Múltiple	varios procesadores
	Unico	un solo procesador
Bases de datos	Significativa	gran uso de base de datos
	Insignificante	escaso o nulo uso de base de datos.

1.2.1.1 Procesamiento concurrente

Esta propiedad expresa el número de eventos manejados por el sistema, y está relacionada con el grado de paralelismo del procesamiento en su interior. El grado de simultaneidad depende de la técnica de realización; es seudo-simultánea cuando un procesador reparte su actividad para satisfacer diversos tratamientos (funcionamiento multitarea). La existencia de instantes precisos en los que el sistema debe responder ante eventos reales introduce una mayor complejidad en el diseño del sistema. Aparecen los problemas de sincronización y exclusión mutua interactuando con imposiciones temporales, lo que complica la capacidad de predicción del comportamiento del sistema debido a las siguientes razones:

- El tiempo de reacción de los componentes que constituyen el sistema solo puede ser realmente evaluado a posteriori, una vez que se ha realizado el diseño y escogido una arquitectura hardware.
- Dada una arquitectura, la política de planificación (scheduling) influye en el comportamiento del sistema.

- Dada una arquitectura y una política de planificación, el reparto de tareas entre los distintos procesadores que constituyen el sistema influye sobre el comportamiento global del sistema.

Debido a estas razones la política de planificación solo puede ser escogida a priori en los casos más sencillos. Según [1] esta es la característica principal para caracterizar los STR.

1.2.1.2 Interfaz Hardware

Habitualmente los STR serán sistemas electrónicos, o con una fuerte componente electrónica, que estarán ligados a su entorno. Este entorno además se considera activo y cambiante. Los sistemas se encuentran en constante interacción mediante sensores y actuadores con el entorno en el que se hallan para controlar su comportamiento. Según el grado de interacción con el entorno dividiremos a los sistemas con Interfaz Hardware significativa o con Interfaz Hardware insignificante.

No se considera en este apartado la comunicación del sistema con el operador, ya que ésta es considerablemente más lenta y va orientada más hacia tareas de monitorización que de control. Por tanto distinguimos entre interfaz hombre-máquina y las interfaces físicas del sistema.

1.2.1.3 Tiempo de reacción ante eventos

Quizás la más característica para muchos autores [2] [3] de las cualidades de los STR es su velocidad de respuesta ante eventos. Los tiempos de intercambios de información en los interfaz hombre-máquina son del orden de décimas de segundo, mientras que los tiempos de reacción de un sistema físico son del orden de ms. Aquí no solo interviene la velocidad del microprocesador, sino que también adquiere gran importancia el resto de los elementos, tales como sensores y actuadores, que componen el sistema.

Por definición [2] se dice que un sistema es de tiempo real cuando efectúa todas sus actividades respetando apremios de tiempo ('*deadline*'). En efecto, la interacción importante con el entorno hace aparecer apremios de tiempo de naturaleza varia, algunas veces muy severos, que se trata de respetar.

- Frecuencia elevada para las solicitudes que provienen del entorno: frecuencia de sucesos, petición de datos.
- Frecuencia elevada para las acciones a emprender relacionadas con el entorno, por ejemplo, frecuencia elevada para el mando de un medio con tiempo pequeño de respuesta.
- Límite máximo para el tiempo de reacción entre el instante de aparición de una solicitud y el del cumplimiento de la acción consecuente (fecha de vencimiento, deadline).

Otras definiciones de STR muy ligadas al concepto del tiempo se pueden encontrar en [4]:

> Un sistema en tiempo real lee entradas de la planta y envía señales de control a la planta en tiempos determinados por consideraciones operacionales de la planta - no en tiempos limitados por la capacidad del computador.

De esta manera es posible definir un programa en tiempo real como:

> Un programa donde el funcionamiento correcto depende tanto de los resultados lógicos y operacionales de los cálculos como del tiempo en el que el resultado es producido.

No respetar alguno de los apremios temporales puede conducir a la aplicación a un estado asimilable al de avería, lo que puede engendrar consecuencias muy graves, para el funcionamiento del sistema e incluso para la vida de las personas. Este es también un hecho diferenciador frente a los sistemas con mucha interfaz hombre-máquina, donde una demora de algún segundo es molesta pero no vital. La forma de evaluar estos sistemas será como rígido (hard) o flexible (soft) según sean graves o leves las consecuencias de no respetar algún apremio.

1.2.1.4 Arquitectura distribuida

Esta característica indica si el sistema puede ser implementado por un único procesador (único) o debe ser distribuido entre varios (múltiples). Estos procesadores pueden estar en la misma tarjeta, en un rac compartiendo bus, acoplados vía red etc. Hay motivaciones, no excluyentes, que llevan a incorporar la interconexión de micros:

- El entorno del sistema: si hay una dispersión geográfica natural, donde para el correcto funcionamiento deben colaborar varias máquinas dispersas, suele requerir que los sistemas de control que manejan cada máquina se comuniquen.

- Prestaciones del sistema: aquí el tiempo de reacción debe ser mínimo y está impuesto por la funcionalidad del sistema, pudiendo ser alcanzado solamente por medio de un procesamiento paralelo.
- Confiabilidad del sistema: se puede exigir una arquitectura distribuida para que en caso de fallo de un procesador no se venga abajo todo el sistema, tomando otros su carga de trabajo, o degradándose paulatinamente.

1.2.1.5 Uso de bases de datos en el sistema

Esta característica hace referencia a la complejidad de los datos, el número de relaciones que deben mantener y la naturaleza de las actualizaciones. Las relaciones complejas entre los datos del sistema pueden complicar la arquitectura del mismo condicionando la metodología utilizada para su desarrollo.

1.2.2 Características secundarias

Son aplicadas a todas las clases de sistemas definidas con las características primarias. Tienden a influir sobre el modo en el que el sistema debería ser desarrollado. Deberán ser tenidas en cuenta como paso previo a la etapa de diseño. Entre la gran cantidad de tales atributos se ha escogido el conjunto mostrado en la siguiente tabla como las más relevantes para los STR.

Característica descriptiva	Conjunto de valores
Fiabilidad	Alta Baja
Reconfigurabilidad	On-line Off-line
Usabilidad (interfaz entre Usuario/computador)	Intuitivo Necesita entrenamiento
Certificabilidad	Esencial No necesaria
Obligaciones (aspectos del entorno)	Significativas Insignificante
Capacidad de evolución	Evoluciona No evoluciona

1.2.2.1 Fiabilidad

Es una propiedad para medir la confianza que se puede depositar en el correcto funcionamiento del sistema. Desde el punto de vista del usuario se puede descomponer en 4 factores: confiabilidad, disponibilidad, seguridad y protección.

- La confiabilidad trata de medir la expectativa de que el sistema preste un servicio ininterrumpido, pocas averías. Ligada a este factor está también la facilidad de mantenimiento del sistema.
- La disponibilidad es una característica que mide las expectativas de un usuario de encontrar el sistema funcionando; depende fuertemente de la confiabilidad y de la facilidad de mantenimiento.
- Seguridad. Describe la propiedad del sistema de no dañar su entorno y provocar accidentes con usuarios u otros elementos.
- Protección. Define la resistencia del sistema ante amenazas externas contra su integridad.

1.2.2.2 Reconfigurabilidad

Con esta propiedad se pretende expresar la posibilidad de que un sistema pueda modificar su estructura y/o funciones durante la operación del sistema. Si un sistema puede cambiar sus propiedades sin degradar la calidad lo podemos catalogar como reconfigurable on-line. Si la modificación de sus propiedades hace necesaria una parada temporal del sistema entonces se dice reconfigurable off-line.

La reconfiguración del sistema puede incluir tanto cambios fundamentales en el funcionamiento o estructura como cambios en los parámetros del sistema.

1.2.2.3 Usabilidad

Se refiere a la facilidad de manejo del sistema por su usuario final. Claramente está relacionado con la complejidad del sistema y con la interfaz usuario-máquina pretendido; muchas veces se debe tener en cuenta si la aplicación final está destinada a gente especializada o no.

1.2.2.4 Certificabilidad

Expresa la posibilidad de obtener garantías de funcionamiento del sistema según las especificaciones. Esta característica depende fuertemente del usuario final e influye en la elección de los métodos de producción del sistema. También implica la definición de una serie de condiciones para certificar el uso adecuado del sistema.

1.2.2.5 Obligaciones

Estas características están directamente ligadas con el funcionamiento externo y con el entorno en el cual va a ser usado. Aquí se hace referencia a todos aquellos elementos del sistema no funcional (el entorno) que no están directamente enlazados con el servicio percibido por los usuarios del sistema. El conjunto de obligaciones está íntimamente ligado con el contexto, es totalmente dependiente del entorno, podría englobar características como el tamaño, forma, consumo de potencia, precio, elección de materiales, rango de temperaturas de funcionamiento etc.

1.2.2.6 Capacidad de evolución

Casi todos los sistemas en tiempo real son capaces de evolucionar hacia otro tipo de sistema. Se trata de ver ahora la facilidad con que un sistema puede modificarse. Por una parte está la necesaria encriptación y seguridad del núcleo, donde solo el suministrador puede efectuar cambios (por ejemplo cambiar una memoria de tipo PROM). Por otra parte es necesaria una flexibilidad para el funcionamiento en su destino final, donde el usuario puede necesitar introducir nuevas reglas durante el tiempo de vida del sistema. En general la estructura del diseño determina la facilidad con que ciertos tipos de cambios pueden hacerse, mientras que la gestión técnica determina quien tiene autoridad para realizar los cambios. Los requisitos deben indicar la dirección de los cambios esperados, para guiar al diseñador en la decisión de qué estructura es la más apropiada.

1.2.3 Características concretas

Conjunto de características que pueden influir en la selección de una forma determinada de desarrollo e implantación del sistema. Suelen ser características impuestas por el cliente.

Algunas de estas características son:

- Plataforma (hardware y software) utilizada en el sistema.
- Tipo de transductores externos y de protocolos de comunicación.
- Sistema replicado (instalación única, o múltiples instalaciones).

A modo de recopilación, en la siguiente tabla aparecen 32 ejemplos, correspondientes a todas las combinaciones que pueden tomar las características definidas como primarias, de diversos sistemas, algunos de tiempo real y otros claramente no.

Característica		Valores permitidos	
PC	Procesamiento concurrente	pocos	muchos
IH	Interfaz hardware	insignificante	significativo
TR	Tiempo de reacción a eventos	flexible	rígido
AD	Arquitectura distribuida	único	múltiples
DB	Base de datos	insignificante	significativo

Clase	Características primarias					Sistema ejemplo
	PC	IH	TR	AD	DB	
0	pocos	insig	flexible	único	insig	Triturador de números (number cruncher)
1	pocos	insig	flexible	único	signi	Catálogo de librería basado en un PC
2	pocos	insig	flexible	multi	insig	Caja registradora en un supermercado
3	pocos	insig	flexible	multi	signi	Vendedor automático en red
4	pocos	insig	rígido	único	insig	Simulador de entrenamiento en tiempo real
5	pocos	insig	rígido	único	signi	Servidor de archivos
6	pocos	insig	rígido	multi	insig	Filtro digital de arquitectura paralela
7	pocos	insig	rígido	multi	signi	Simulador de vuelo de entrenamiento: Sería el simulador de la clase 4 más todo un entorno, extraído de una base de datos. La interfaz humana implica pocas interacciones, pero el tiempo de reacción es crítico
8	pocos	signi	flexible	único	insig	Control de ambiente: (Calefacción, aire acondicionado). Los parámetros de control, dados por sensores , como humedad y temperatura se regulan de acuerdo a órdenes del usuario. Aparecen pocas interacciones concurrentemente y la perdida de un dato no implica fallo en el sistema
9	pocos	signi	flexible	único	signi	Sistema de medida: Toma de datos periódica de sensores, archivándoles en ficheros junto con información estadística.
10	pocos	signi	flexible	multi	insig	Sistema de adquisición de datos meteorológicos:. Toma datos en lugares dispersos, realiza cálculos y los envía a puestos de cálculo centralizados.
11	pocos	signi	flexible	multi	signi	Sistema de adquisición de datos metereológicos con base de datos: Análogo al anterior pero registrando los datos en una base, para su posterior análisis.
12	pocos	signi	rígido	único	insig	Ascensor: Aplicación para controlar el movimiento del ascensor y garantizar su seguridad. Hay pocas interacciones concurrentes pero es necesario reaccionar apropiadamente, especialmente cuando se detiene.
13	pocos	signi	rígido	único	signi	Sistema de seguimiento por radar: Toma datos periódicamente de un radar y se los proporciona a un sistema para el seguimiento de aviones. A partir de los datos recogidos es posible identificar el avión comparándolos con los almacenados en una base de datos.
14	pocos	signi	rígido	multi	insig	Sistema de control de freno: En los aviones se trata de un sistema multiprocesador que debe reaccionar tanto a las ordenes del piloto como al contacto de las ruedas con la pista de aterrizaje. Este sistema debe responder rápida y correctamente frente a eventos externos tomados periódicamente de sensores hardware.
15	pocos	signi	rígido	multi	signi	Sistema de seguimiento por radar distribuido: Similar al tipo 13 pero con varios radares intercomunicados.

16	muchos	insig	flexible	único	insig	Sistema de control de accesos automático: Compuesto de varias puertas compartiendo todas ellas la misma unidad de control
17	muchos	insig	flexible	único	signi	Sistema de información meteorológica y aereonaútica: Sistema para informar a los aviones en vuelo dándoles información meteorológica y ayudas a la navegación.
18	muchos	insig	flexible	multi	insig	Correo electrónico: Sistema distribuido con una interfaz con el operador humano de tipo flexible sin operar con base de datos significativamente.
19	muchos	insig	flexible	multi	signi	Reserva de billetes para aviones: Sistema distribuido con interfaz con el operador y significativo uso de base de datos
20	muchos	insig	rígido	único	insig	Sintetizador electrónico: Maneja muchos eventos a través de su teclado en un contexto de cortos tiempo de reacción.
21	muchos	insig	rígido	único	signi	Sistema de control de bolsa: Acepta pedidos, los registra y calcula en un severo tiempo real la evolución de los precios del mercado.
22	muchos	insig	rígido	multi	insig	Sintetizador musical multiprocesador: Como en la clase 20 pero incorporando varias voces.
23	muchos	insig	rígido	multi	signi	Sistema de control de bolsa distribuido: Como en la clase 21 pero distribuido
24	muchos	signi	flexible	único	insig	Sistema de monitorización de tráfico local
25	muchos	signi	suave	único	signi	Sistema de gestión de componentes: Almacena datos de todos los componentes y recoge sus llegadas y partidas en una base de datos. Puede repartir los componentes disponibles cuando se soliciten en los talleres por medio de sistemas electromecánicos controlados por el sistema
26	muchos	signi	suave	multi	insig	Pantalla para la información del tráfico: Recibe información acerca del estado del tráfico desde diferentes lugares e informa a los conductores con los mensaje apropiados.
27	muchos	signi	suave	multi	signi	Sistema de control del tráfico: Recibe el estado del tráfico desde varios lugares para establecer, en tiempo real, la estrategia óptima para la temporización de los semáforos, avisos de rutas alternativas etc.
28	muchos	signi	rígido	único	insig	Sistema de tarjetas hardware de interfaz E/S: Controla la adquisición de datos de múltiples dispositivos para suministrar a las correspondientes aplicaciones los servicios de E/S, aquí hay una fuerte dependencia temporal.
29	muchos	signi	rígido	único	signi	Sistema de recogida de eventos en tiempo real: Capta todos los eventos significativos en una aplicación de tiempo real en una base de datos. Se emplea para depurar el sistema o grabar información para un análisis posterior. No se deben perder eventos y no debe interferir con el proceso operacional.
30	muchos	signi	rígido	multi	insig	Sistema de control de un vehículo: Estos sistemas tienen distribuidos los controles para cada una de las funciones (frenado, guiado, encendido del motor etc.) Todos estos sistemas deben responder rápidamente ante cualquier demanda.
31	muchos	signi	rígido	multi	signi	Estación de supervisión de procesos: Monitoriza el control de procesos en un ambiente industrial distribuido.

Para nosotros esta clasificación solo tiene un carácter orientativo y es ilustradora de la diversidad de sistemas con que un diseñador se puede encontrar, atendiendo solamente a las características aquí definidas como primarias. Por supuesto buena parte de los sistemas anteriores no pueden ser considerados como de tiempo real (según [1] las clases 0 a 5 quedan fuera del alcance de los STR). El resto de clases se puede considerar que, en mayor o menor medida, sí son STR.

1.3 Elementos de un sistema de control por computador

A modo de ejemplo simple, ilustrativo de las operaciones de un sistema de control por computador, vamos a considerar el calentador de aire de la figura 1.1. Un ventilador produce una corriente de aire sobre un elemento calentador situado en el interior de un tubo. Al final del tubo, un dispositivo cierra el circuito y suministra un voltaje proporcional a la temperatura (B). El voltaje responsable de la corriente suministrada al elemento calentador puede ser variado en el punto (A). La posición de la cubierta de entrada de aire en el ventilador se ajusta mediante un motor reversible. El motor funciona a velocidad constante y se enciende y apaga (on/off) mediante una señal lógica. Una segunda señal lógica determina la dirección de rotación. Un potenciómetro conectado a la cubierta da un voltaje proporcional a la posición de la misma.

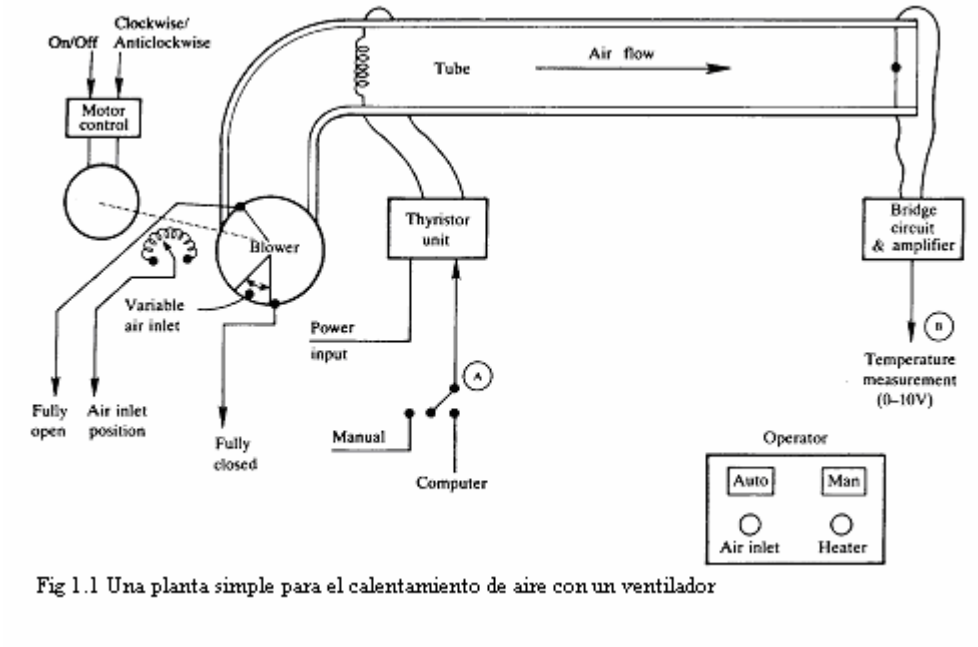

Fig 1.1 Una planta simple para el calentamiento de aire con un ventilador

Desde un panel de control, el operador puede seleccionar el tipo de control, manual o automático. En modo manual, el operador puede ajustar la posición de la cubierta y el elemento calentador. La utilización de un computador en la operación de esta planta requiere del software para monitorización, control y actuación de la planta. La figura 1.2 muestra un esquema general del sistema.

Monitorización implica la obtención de información acerca del estado actual de la planta. En nuestro ejemplo, esta información se obtiene de dos formas:

- Señales analógicas: Temperatura del aire, posición de la cubierta.
- Señales digitales: Cubierta totalmente abierta/totalmente cerrada, motor on/off, calentador on/off, manual/automático.

Control. El computador se encargará de controlar la temperatura de salida (control con realimentación) y la posición de la cubierta del ventilador. Son necesarias también operaciones de secuenciamiento e interbloqueo, puesto que, por ejemplo, el calentador no debería estar encendido si el ventilador no está funcionando.

La actuación requiere la provisión de un voltaje proporcional al calor que ha de proporcionar el calentador; señales lógicas indicando on/off y el sentido de giro de la cubierta; y señales lógicas para la pantalla del operador.

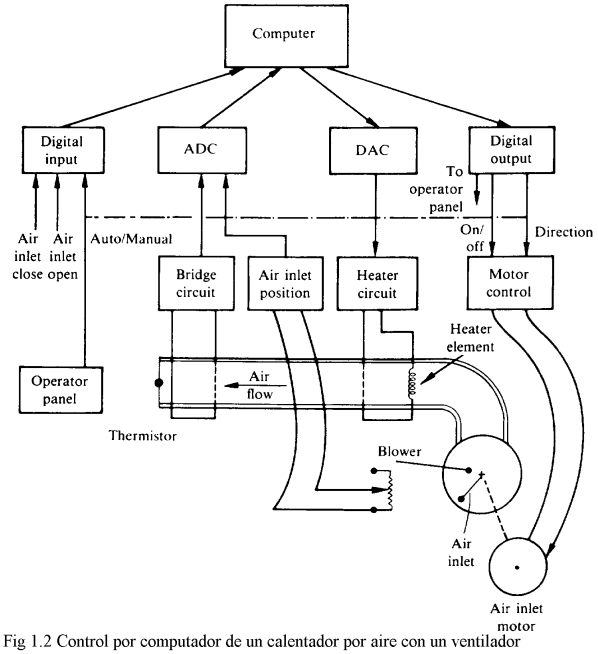

Fig 1.2 Control por computador de un calentador por aire con un ventilador

Las tareas de monitorización y actuación conllevan una serie de dispositivos de interfaz, entre los que se incluyen convertidores A/D y D/A, líneas de input/output digital, etc. Cada uno de los dispositivos requerirá del software necesario para su operación.

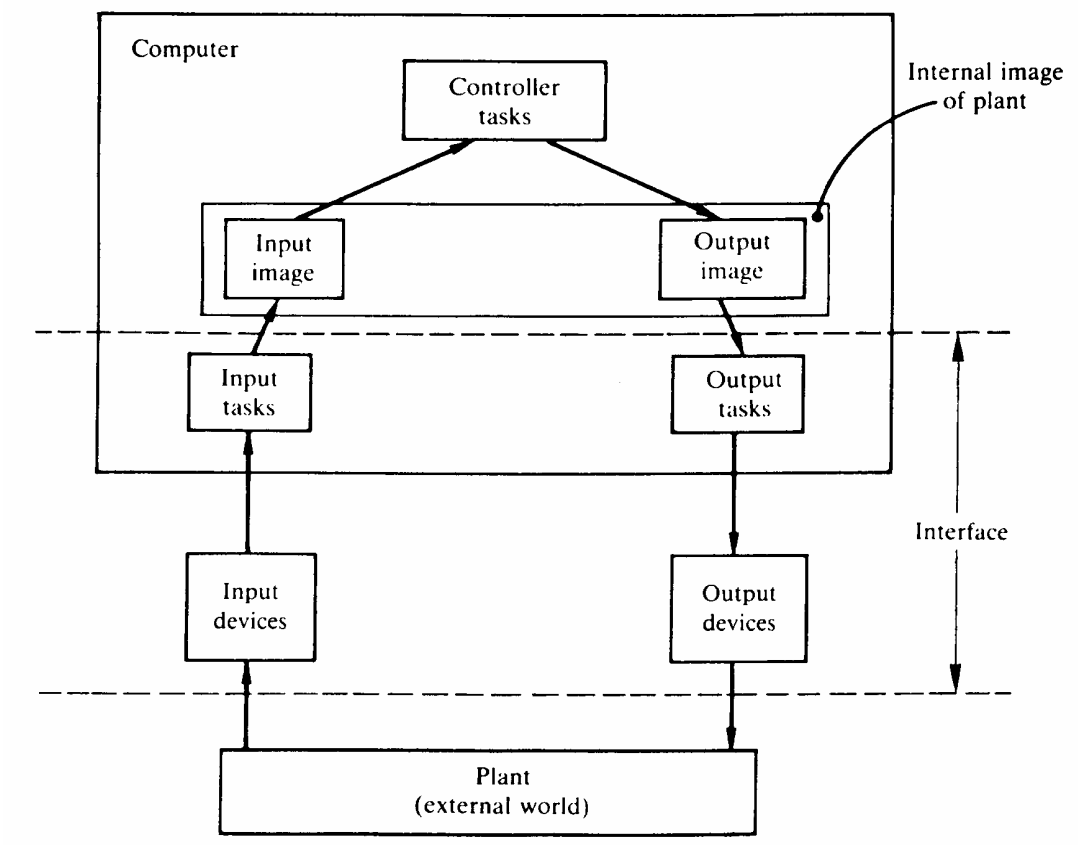

Fig 1.3 Control por computador generalizado mostrando interface software y hardware

Los dispositivos de input junto con el software de input forman la 'imagen input' de la planta (ver figura 1.3). Esta imagen representa el estado de la planta en un instante, estado que es renovado a intervalos prefijados. El proceso de obtención de esta imagen implicará digitalización (conversión de una medida continua en un valor discreto) con un determinado periodo de muestreo.

La imagen output representa el conjunto de salidas generadas por los cálculos en el computador. Esta imagen es actualizada periódicamente por el software de control, cuya tarea será operar sobre la imagen input, mediante los algoritmos de control adecuados, para la obtención de la imagen output. El software de output se encargará de enviar los datos contenidos en la imagen output hacia la planta, a través de los dispositivos de salida.

La figura 1.4 representa un diagrama de bloques simplificado del bucle de control de la temperatura de nuestro sistema. Ts representa el periodo de muestreo. Si, por ejemplo, le damos el valor de 10ms, cada 10 ms el computador debe leer los valores de input, realizar los pertinentes cálculos de control para obtener la imagen output, y enviar dicha imagen hacia la planta, a través de los dispositivos de salida.

Como hemos visto, las tareas de software se pueden dividir en tres grandes áreas:

Tareas de input de la planta.

Tareas de output de la planta.

Tareas de control.

El control de un sistema puede estar compartido entre diversos computadores, situados posiblemente en sitios diferentes. Debemos por tanto añadir las tareas de comunicación a las tres anteriormente enumeradas. Estas tareas se encargarán de las operaciones de input/output con dispositivos de almacenamiento, impresoras, redes de área local, etc.

El computador puede operar con las tareas software mencionadas en modo secuencial, repitiendo la secuencia indefinidamente, o más comúnmente, en modo concurrente (paralelo o seudoparalelo, según se trate de un sistema mono o multiprocesador).

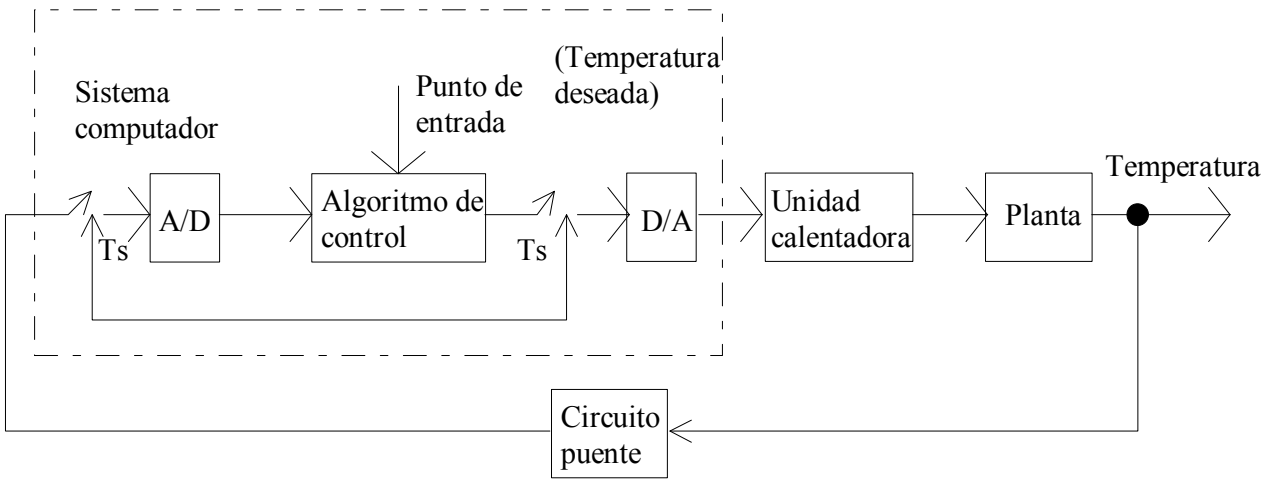

Figura 1.4 Diagrama de bloques del bucle de control

1.4 El tiempo en los STR

Una característica común de los STR, como ya quedó antes señalado, es que el computador forma parte del sistema con el cual interactúa. Las señales tomadas por sensores, las enviadas a los actuadores, las comunicaciones de las tareas, están conectadas por dispositivos físicos a procesos que son externos al computador. Estos procesos

externos operan en su *propia escala de tiempos* y se dice que el computador funciona en tiempo real si las acciones que realiza están relacionadas con la escala de tiempos de esos procesos externos. [4]

La sincronización entre los procesos externos y las acciones internas (tareas) llevadas a cabo por el computador puede definirse en términos de tiempo transcurrido desde que se producen las acciones o también en términos de la hora del día. En estos casos se dice que el sistema está basado en reloj y las operaciones se realizan según una planificación horaria. Pero también puede definirse un sistema en términos de eventos, por ejemplo el cierre de un conmutador, en cuyo caso se dice que está basado en eventos.

Hay una tercera categoría, interactiva, en la cual la relación entre las acciones de el computador y el sistema están mucho menos definidas. En este rango caen típicamente las aplicaciones donde los requisitos son que un conjunto de operaciones en el computador deben estar completadas dentro de un tiempo predeterminado. La mayoría de las tareas de comunicación caen dentro de este rango. Aunque sin unos conocimientos de control no resulta evidente, las tareas de control también están directamente conectadas al entorno, siendo necesario en muchas ocasiones trabajar en tiempo real, dado que el tiempo está envuelto en la determinación de los parámetros usados en el algoritmo de un control digital. Atendiendo a la relación del computador con su entorno y el tiempo en [4] se establece una nueva clasificación.

1.4.1 Tareas basadas en el reloj (cíclicas, periódicas)

El tiempo necesario para la respuesta de una planta, sus constantes de tiempo, varía enormemente según la naturaleza de la misma. Puede ser del orden de ms en un sistema electrónico o de horas en uno químico. En un lazo de control típico, donde es necesaria una realimentación de las salida por medio de un proceso de muestreo, se tiene que a menor constante de tiempo del sistema mayor debe ser la velocidad de muestreo empleada, y a la inversa. El computador, si funciona en tiempo real, debe ser capaz de enviar todas las señales de salida tras efectuar los cálculos precisos (medida, control y actuación) dentro de cada intervalo de muestreo.

La finalización de las tareas de control dentro del tiempo establecido dependerá del número de operaciones a realizar y de la velocidad con que pueda ejecutarlas el computador. La sincronización es obtenida habitualmente añadiendo un reloj al procesador (conocido normalmente como reloj en tiempo real) y usando una señal generada por este reloj para *interrumpir* las operaciones del computador en instantes de tiempo predeterminados. En sistemas grandes las tareas pueden subdividirse en grupos para controlar diferentes partes de la planta, pudiendo ocurrir que cada parte tenga su propio intervalo de muestreo.

1.4.2 Tareas basadas en eventos (aperiódicas)

Hay muchos sistemas donde las acciones no deben realizarse en instantes de tiempo predeterminados, o en intervalos de tiempo ya establecidos, sino que deben responder ante determinados eventos. Un ejemplo típico podría ser el cierre de un conmutador para detener una bomba cuando el nivel de líquido llega al límite preestablecido en un depósito. Este tipo de sistemas están muy extendidos cuando se debe iniciar una acción ante una señal de alarma. En estos casos el requisito temporal es que el sistema debe responder antes de un tiempo desde el momento en que se ha producido la señal de alarma.

Los sistemas basados en eventos también emplean normalmente las interrupciones para indicar al computador la aparición de la señal de alarma, aunque en algunos casos muy simples pueden usares técnicas de *polling*, donde el computador se dedica a ir estudiando si se ha producido una señal de alarma *preguntando* a los sensores si se ha producido alguna señal.

1.4.3 Sistemas interactivos

Conforman un amplio rango de sistemas, cuyas exigencias suelen manifestarse de forma que el tiempo promedio de respuesta no debe exceder de escasos segundos (o décimas de segundo). Un ejemplo de este tipo de sistemas podría ser un cajero automático. Se pueden diferenciar de los sistemas basados en eventos en que, a pesar de responder frente a eventos externos, lo hace dependiendo del estado interno del computador y sin *ninguna referencia al entorno*. Muchos sistemas interactivos pueden dar la impresión de que están basados en reloj, puesto que ofrecen funciones horarias y de calendario, eso se debe a que incorporan un reloj de tiempo real que ofrece estos servicios.

1.5 Clasificación de los programas

La importancia de separar las actividades llevadas a cabo por el computador entre tareas no tiempo real y tareas tiempo real, y la división de estas últimas en dos tipos diferentes, viene dada por los diferentes niveles de dificultad en el diseño e implementación de los distintos tipos de programas. Estudios experimentales [4] demuestran que es más difícil construir programas para ejecutar tareas en tiempo real y operaciones de interfaz que programas de procesamiento de datos. La división del software en pequeños y coherentes módulos es una técnica básica, dichos módulos muchas veces estarán ligadas a constantes de tiempo diferente. Algunos autores identifican tres tipos de programación:

Secuencial.

Multitarea.

Tiempo real.

1.5.1 Secuencial

En la programación secuencial clásica las acciones están estrictamente ordenadas como secuencias de ordenes: el comportamiento del programa depende solo de los efectos de las acciones individuales y su orden, el tiempo necesario para realizar la acción no tiene importancia en el resultado final. La verificación de este tipo de programas es sencilla, basta hacer una ejecución paso a paso, ya que una sentencia implica un acción, que siempre será igual.

1.5.2 Multitarea

Los programas multitarea difieren de los clásicos programas secuenciales en que las acciones necesarias para su realización no son necesariamente disjuntas en el tiempo, pudiendo ser necesario que varias acciones se realicen en paralelo (o pseudo-paralelo). Los programas se construyen en partes (tareas) cada una de ellas puede ser totalmente secuencial, pero que se ejecutan concurrentemente y se comunican a través de variables compartidas y señales de sincronización. En este caso las tareas se pueden verificar por separado sólo si las variables que manejan no son compartidas. En el caso contrario las interacciones de las tareas pueden hacer impredecible el efecto del programa, a menos que haya alguna regla más que gobierne el secuenciamiento de las acciones de las tareas.

1.5.3 Tiempo real

En un programa en tiempo real, además de las características de los dos anteriores, se deben considerar la aparición de eventos que dependen del entorno, no del diseñador del programa. Es decir, dichos eventos pueden aparecer durante cualquier operación del computador y no pueden manejarse según las reglas de sincronización intertareas. Un programa en tiempo real puede seguir dividiéndose en tareas, pero la comunicación entre tareas puede no esperar necesariamente por una señal de sincronización, ya que la tarea del entorno puede no admitir retardos.

BIBLIOGRAFÍA

En la elaboración de estos apuntes se han consultado los capítulos introductorios de los siguientes libros:

[1] I. Pyle, P. Hruschka, M.Lissandre.. 'Real Time Systems. Investigating Industrial Practice.' Ed Wiley 93 ISBN 0-471-93553-0

[2] Jean Paul Calvez 'Especificación y Concepción de Sistemas. Una Metodología'. Publicado en la escuela IRESTE (Nantes) 90

[3] M. Schiebe, S. Pferrer 'Real Time Systems Engineering and Applications' Ed Kluwer 92 ISBN 0-7923-9196-9

[4] Stuart Bennet. 'Real Time Computer Control' Ed Prentice Hall 94 ISBN0-13-764176