2. Técnicas de medida y de presentación de los resultados

2.1. Herramientas de medida: Monitores

2.1.1. Introducción

A la hora de realizar cualquier tipo de estudio sobre un sistema informático (ampliación, utilización, sintonización, etc.), es necesario tener información real acerca de lo que está ocurriendo en ese sistema. Para ello, es de vital importancia disponer de herramientas que permitan obtener esa información. Estas herramientas son conocidas con el nombre de monitores.

Un *monitor* es una herramienta utilizada para observar la actividad de un sistema informático mientras es utilizado por sus usuarios. En general, observan el comportamiento del sistema, recogen datos estadísticos de la ejecución de los programas, analizan los datos recogidos y presentan los resultados.

Una característica de una medida debe ser su repetibilidad, esto es, la posibilidad de realizar una medición diferentes veces, y que el resultado de este proceso (la medida) sea siempre el mismo. En caso de que no sea así, las sucesivas medidas se usan para reducir los errores intrínsecos del proceso de medición.

Ahora bien, en informática, el resultado de una medición será distinto unas veces de otras, ya que, normalmente, no es posible repetir las mismas condiciones de carga y en los mismos instantes. Por ello, se habla de monitorización y no de medición ya que lo que estrictamente se efectúa es un seguimiento de la actividad realizada.

El desarrollo de tales instrumentos para el seguimiento del comportamiento de sistemas informáticos utiliza las dos tecnologías que conviven en un sistema informático: el hardware y el software.

Otro objetivo de una herramienta de medida es cuantificar los resultados de una observación.

Los monitores pueden ser utilizados por todos los usuarios del sistema, desde los analistas de sistemas hasta los gestores y programadores del mismo.

La información aportada por el monitor puede ser útil:

— Para el usuario y administrador, pues puede ser necesario conocer una serie de características del sistema (capacidad, posibilidad de ampliación, planificación, etc.).
  - Un administrador de sistema puede conocer la utilización de los recursos del sistema o el cuello de botella del mismo.
  - Un programador puede encontrar los segmentos de software más utilizados y, en consecuencia, puede mejorar el comportamiento de los programas.
  - Un administrador puede ajustar los parámetros del sistema y mejorar las prestaciones del mismo.
  - Un analista de sistemas puede caracterizar la carga y crear cargas de prueba, cuyos resultados se pueden utilizar para planificar la carga del sistema.
  - Un analista de sistemas puede encontrar los parámetros de un modelo y tener datos más reales para los modelos que representan el comportamiento temporal del sistema.
— Para el propio sistema, en aquellos que permiten su adaptación dinámica con la carga.

2.1.2. Características

Se ha dicho que el objetivo de un instrumento de monitorización consiste en cuantificar los resultados de la observación de un sistema informático. El desarrollo de herramientas para observar el comportamiento de un sistema informático sometido a una carga ha seguido dos direcciones separadas, caracterizadas por las tecnologías que se usan en un sistema informático. Los monitores software pueden ser programas, que detectan estados del sistema, o conjuntos de instrucciones, que se podrían denominar sondas software, capaces ambos de detectar acontecimientos. Los monitores hardware son dispositivos electrónicos que deben conectarse a puntos específicos del sistema (mediante sondas electrónicas) para detectar señales (niveles o impulsos de tensión) que caracterizan los fenómenos que deben observarse.

Aunque los monitores no se utilizan estrictamente como instrumentos de medida, su calidad viene definida por las mismas características, es decir:

— *Sobrecarga o interferencia.* Como todas las herramientas que miden fenómenos físicos, los instrumentos que observan el comportamiento de los sistemas informáticos, sustraen energía del sistema que están midiendo. La energía consumida por el instrumento de medida debe ser tan poca como sea posible de forma que la perturbación introducida por el instrumento no altere los resultados de la observación. En lo concerniente a los monitores hardware este peligro existe a través de los puntos de conexión; no obstante, el uso de sondas de muy alta impedancia puede hacer despreciable esta perturbación. Por otro lado, los monitores software aumentan la carga del sistema y alteran, por consiguiente, su comportamiento. Sin embargo, permiten la obtención de medidas inalcanzables por un monitor hardware (aunque lo contrario también es cierto) y presentan ciertas ventajas de tipo operativo, por lo que se intentará reducir al mínimo posible los efectos de su presencia.
— *Precisión.* Otra característica de las herramientas de medida es su exactitud, la cual puede expresarse por el error que puede afectar al valor de los datos recogidos. Estos errores son debidos a diferentes causas: la interferencia del propio monitor, a una incorrecta instalación o utilización, al número de dígitos para representar la medición, etc. Cuando se utiliza un conjunto de herramientas cada una de ellas aporta su inexactitud al error total. Normalmente, este coeficiente de error debería ser adjuntado por el fabricante, pero no lo suele dar, siendo el usuario quien debe evaluarlo, siempre que sea posible, por contraste de la herramienta.
— *Resolución.* Es la máxima frecuencia a la que se pueden detectar y registrar correctamente los datos. Es decir, mide la capacidad de la herramienta de separar dos acontecimientos consecutivos en el tiempo.
— *Ámbito o dominio de medida.* Hace referencia al tipo de acontecimientos que puede detectar y, en consecuencia, a aquellas características que es capaz de observar y medir.
— *Anchura de entrada.* Es el máximo número de bits de información de entrada que el monitor puede extraer en paralelo y procesar cuando se produce un acontecimiento.
— *Capacidad de reducción de datos.* Es la capacidad que puede tener el monitor de analizar, procesar y empaquetar datos durante la monitorización para un mejor tratamiento y comprensión de los mismos y para reducir el espacio necesario para almacenar los resultados.
— *Compatibilidad.* El hardware y el software de monitorización deben ser fácilmente adaptables a cualquier entorno y a cualquier requerimiento de la aplicación. Las herramientas no deben ser específicas para una determinada aplicación o sistema. Durante la operación de monitorización, la herramienta debe poder adaptarse de forma interactiva a las necesidades que puedan ir presentándose.
— *Coste.* Como en cada equipo industrial, el coste es un factor muy importante a tener en cuenta. Hay que considerar no sólo el coste de adquisición sino también los de instalación, mantenimiento, formación y operación.
— *Facilidad de instalación.* Otro factor a tener en cuenta es la facilidad de instalación del monitor, y también la facilidad de retirarlo del sistema, en caso de no ser necesario.

— *Facilidad de utilización.* Las herramientas de monitorización y medida deben ofrecer al usuario una interfaz que pueda ser utilizada por cualquier programador y no sólo por personal experto. El usuario debe poder centrar fácilmente su interés de forma dinámica desde cuestiones de alto nivel, como paralelismo, mensajes, etc., hasta otras de más bajo nivel o más puntuales, como procesos en espera, tiempos de retardo, etc.

Otros requerimientos típicos de estas herramientas, que es necesario tener en cuenta a la hora de realizar o seleccionar un monitor, son los siguientes:

— *Monitorización continua durante la operación.* Considerando un sistema distribuido que cambia continuamente por la creación y extinción de procesos y por la modificación de la estructura de interconexión entre las unidades distribuidas, es necesario tener un monitor que refleje la secuencia de modificaciones y la utilización de las unidades.
— *Presentación orientada a la aplicación.* La gran cantidad de datos de monitorización y prestaciones debe presentarse de forma orientada a la aplicación, reflejando la organización y la semántica del programa, dando los resultados preferiblemente en forma de tablas y gráficos. El almacenaje eficiente de la información puede aportar detalles acerca del pasado en caso de errores o comportamiento anormal del programa, a la vez que presentar resultados más detallados para analizar los problemas de prestaciones y diseño.
— *Integración.* La instrumentación debe ser transparente a los usuarios, debiendo permitir la realización de una monitorización continua del sistema sin una especial preparación. En general, la información acerca del sistema debe estar integrada en éste, debiendo concebirse esta integración en fase de diseño del monitor.
— *Realimentación.* En algunos casos, para conseguir un comportamiento adaptativo del sistema, los resultados de la monitorización deben ser realimentados al sistema para mejorar su comportamiento, como, por ejemplo, para algoritmos de equilibrado de la carga.

2.2. Conceptos de medida

Cuando se está frente a fenómenos de naturaleza compleja, que resultan de la coexistencia e interacción de muchos factores, se necesita describir con tanta precisión como sea posible lo que se está observando, puesto que se deben explorar las conexiones entre los acontecimientos para que sea posible explicarlos y predecirlos. Un método vital para alcanzar este objetivo es el de la medida, que permite tratar cuantitativamente los fenómenos; es decir, permite sugerir, verificar y establecer las relaciones entre las cantidades que caracterizan un fenómeno.

El estudio de un sistema informático, tanto cuando se está diseñando, como cuando está trabajando, requiere el uso de técnicas de medida, que son particularmente útiles en la determinación del comportamiento del sistema. ¿Qué significa medir (o, mejor, monitorizar) un sistema? En general, significa recoger información de la actividad del sistema mientras atiende a los usuarios.

Las medidas (u observaciones) pueden clasificarse en dos grandes categorías, atendiendo a quien las va a usar: las requeridas por los usuarios del sistema y las requeridas por el propio sistema. Todas las medidas concernientes a la utilización de los recursos del sistema llevadas a cabo para evaluar su comportamiento, controlar su uso y planificar la adición de nuevos recursos pertenecen a la primera clase. Las efectuadas para el propio control del sistema, que le permiten adaptarse dinámicamente a los factores que condicionan su actividad (principalmente la carga de trabajo) y mantener un nivel adecuado de comportamiento externo, pertenecen a la segunda clase.

En general, hay que tener en cuenta que ni el hardware ni el software de un sistema informático se han diseñado con el propósito de ser medidos (u observados) al menos en sus primeros tiempos; actualmente esta concepción ha cambiado desde que se ha asumido la necesidad de conocer el comportamiento de un sistema informático. Este defecto de concepción, a menudo, restringe las medidas (u observaciones) que pueden realizarse de un sistema y requiere que se modifiquen algunas de sus funciones o que se añadan instrumentos caros para sobrepasar dichas restricciones.

La mesurabilidad de un sistema informático se puede definir como una función de la información que se puede obtener con un monitor y del coste de las mediciones. Es decir, la mesurabilidad de un sistema puede variar entre dos extremos, correspondiendo a la posibilidad de medir (observar) cada componente individual del sistema con el nivel de detalle deseado y a la total inaccesibilidad al sistema, respectivamente.

Hay diversas técnicas para medir (observar) un sistema informático; su elección depende del tipo de análisis que deba efectuarse y del nivel al que deba realizarse. Puesto que un acontecimiento es un cambio del estado del sistema, una forma de recoger datos de determinadas actividades del sistema es capturar todos los acontecimientos asociados a los cambios de estado y registrarlos en el mismo orden en que se producen. En este caso, la medición (observación) se efectúa por detección de acontecimientos.

Una técnica que, con frecuencia, se prefiere a la que se acaba de proponer, ya que no interfiere tanto el comportamiento del sistema, es la del muestreo, que consiste en interrumpir el sistema a intervalos regulares o aleatorios para detectar el estado de alguno de sus componentes. Si el número de muestras recogidas es suficientemente grande, este tipo de medidas puede ser suficientemente preciso.

2.2.1. Detección de acontecimientos

Se ha definido un acontecimiento como un cambio de estado del sistema. Se puede definir el estado de un sistema informático como el establecido por los

valores de todas sus memorias (en un sentido amplio, es decir, todo soporte capaz de almacenar alguna información visible o no desde el nivel del programa). Evidentemente, el seguimiento del estado de todas ellas constituiría una tarea ímproba que requeriría un sistema tanto o más grande que el que se está observando. Cuando se observa un sistema con ánimo de evaluar su comportamiento, normalmente se restringe el estado del sistema a un conjunto limitado de memorias del sistema, que englobe aquellas que reflejen los cambios en el comportamiento del sistema que interesa retener, de acuerdo con los objetivos del estudio que se hayan planteado.

Cuando un acontecimiento está asociado con una función de un programa se dice que se está frente a un acontecimiento software, como, por ejemplo, el hecho de iniciar un programa una operación de E/S. De forma similar un acontecimiento hardware consiste en la aparición de una o más señales en los circuitos de un componente del sistema y es independiente del programa que se esté ejecutando en ese momento, como, por ejemplo, los movimientos del brazo de un disco.

No obstante, muchos acontecimientos hardware pueden ser reconocidos por el software ya que van acompañados de la modificación de alguna memoria accesible al software, es decir, llevan asociados un acontecimiento software. En la discusión posterior, cuando se efectúan medidas (observaciones) por detección de acontecimientos, se supondrá que se trata con acontecimientos reconocibles por el software. Por consiguiente, se ilustrará esta técnica haciendo referencia a acontecimientos software, aunque, conceptualmente se puede usar en medidas hardware.

El principio de detección de acontecimientos software es el de insertar un código suplementario (*traps*, es decir, interrupciones controladas por el programa) en lugares determinados del sistema operativo. Cuando se produce un acontecimiento que debe detectarse, este código transferirá el control a la rutina de tratamiento, que almacenará sus datos significativos, junto con el instante de aparición, en un área tampón, para posteriormente grabarlos en disco y devolver el control al sistema operativo. El conjunto de todos los datos registrados de esta forma constituye lo que se denomina una traza de acontecimientos. No obstante, hay que ser prudente a la hora de seleccionar la cantidad de acontecimientos que se desea registrar, ya que se corre el riesgo de que la observación de los acontecimientos software provoque una distorsión notable en el comportamiento del sistema.

En sistemas conducidos por interrupciones, se puede adoptar una técnica ligeramente distinta para detectar la aparición de determinados acontecimientos. En este caso, la dirección de la rutina de tratamiento de una interrupción determinada debe sustituirse por la dirección de inicio de la rutina de medición. Esto permite interceptar cada interrupción de este tipo y leer, cuando se produce la interrupción, los contenidos de determinadas posiciones de memoria o tablas. Estos datos, junto con el instante de aparición, se registrarán en cinta o

disco antes de devolver el control al sistema operativo para el tratamiento de la interrupción.

Algunos sistemas, normalmente grandes, no obstante, están equipados de hardware especial para facilitar la captación de acontecimientos por parte del software y disminuir su coste, especialmente en necesidades de CPU.

2.2.2. Muestreo

La técnica de detección de acontecimientos tratada en el apartado anterior se basa en la intercepción y registro de todos los acontecimientos de tipos determinados. Se trata, pues, de una técnica que se deberá utilizar sólo cuando sea necesario conocer la secuencia de estos acontecimientos o el número de veces que se han producido en un intervalo determinado. Cuando el conocimiento de estos elementos no es esencial para el estudio de un problema, puede adoptarse el método de muestreo.

El *muestreo* es una técnica estadística que se puede usar siempre que el conocimiento de todos los componentes, objetos o acontecimientos de una población sea imposible, poco práctica o demasiado cara. Entonces, en vez de examinar todos los elementos de la población completa, este método analiza solamente una parte de ellos, denominada una muestra. A partir de esta muestra es posible estimar, normalmente con un grado de precisión elevado, algunos de los parámetros estadísticos que caracterizan la población.

En la evaluación del comportamiento de un sistema informático, esta técnica presenta la ventaja de producir un volumen de datos mucho menor, reduciendo y simplificando, por lo tanto, su análisis. Además, el proceso de recogida de datos causa unos disturbios inferiores al sistema y esta reducción de interferencia hace que la alteración de la evaluación del comportamiento debida al instrumento de medida (observación) sea a menudo irrelevante y siempre más fácilmente controlable.

El muestreo puede usarse con dos objetivos distintos:

1. Evaluar las fracciones de un intervalo de tiempo dado que cada componente del sistema ha permanecido en distintos estados. Los datos recogidos durante el intervalo de medida están sujetos a un análisis posterior para determinar qué sucedió durante dicho intervalo.
2. Seguir la evolución de un sistema y predecir su comportamiento futuro de forma que puedan tomarse las decisiones que tendrán una influencia positiva en su comportamiento.

El primer objetivo se alcanza con herramientas de medida específicas, utilizadas normalmente por los usuarios de un sistema para evaluar la eficiencia de uso de los recursos. La precisión de los resultados viene determinada por el tamaño de la muestra. Debe observarse que, puesto que las cantidades muestreadas son función del tiempo, será necesario suponer que la carga es estacionaria durante largos períodos de tiempo.

No obstante, pueden obtenerse situaciones razonablemente estacionarias subdividiendo el intervalo de medida en períodos de tiempo relativamente cortos (de fracciones de minuto a algunos minutos) y agrupando los bloques de datos homogéneos.

El segundo objetivo se alcanza cuando la medida es una parte integrante del proceso de control de la actividad de un sistema informático. Muchas de las funciones del sistema operativo deben adaptarse dinámicamente a las variaciones de la carga o, de forma más general, a las variaciones de las condiciones del sistema. Las estimaciones de los parámetros, sobre las que se basan determinadas decisiones, consideran la historia reciente de sus valores. Puesto que la evolución del sistema no puede predecirse de forma determinista, es necesario crear un modelo basado en la estructura estocástica subyacente. En particular, en un sistema informático, las variables que representan los estados de los componentes del sistema son constantes por intervalos con saltos en instantes aleatorios; es decir son variables aleatorias discretas. No obstante su importancia, este tipo de modelos no se considerará en este texto ya que tienen mayor influencia en el diseño de sistemas que no en la evaluación del comportamiento.

2.3. Estructura del monitor

El esquema conceptual del monitor de un sistema computador es el que aparece en la figura 2.1.

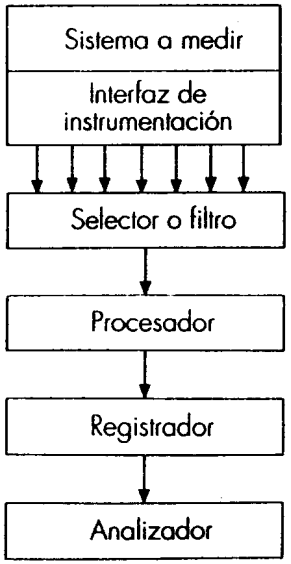

FIGURA 2.1.

La conexión entre el monitor y el sistema se realiza a través de la interfaz de instrumentación. Esta interfaz incluye todos aquellos elementos que permiten acceder a los puntos del sistema que contienen información relevante para la monitorización, tales como, por ejemplo, secciones de código incluido en el sistema operativo, temporizadores, puntos de sondeo, etc.

El elemento selector o de filtrado permite una captura selectiva de las actividades sondeadas. Esto le permite al usuario orientar la medida hacia los aspectos específicos que desea explorar. Además tiene por misión generar aquellas variables que no teniendo existencia física directa en el sistema, pueden generarse en función de otras que son directamente observables.

El elemento procesador realiza las comprobaciones de los elementos del sistema que van a ser analizados o medidos. El resultado de las comprobaciones se graba en un medio de almacenamiento para poder ser analizado e interpretado y así obtener los resultados deseados.

Algunas veces la fase de análisis e interpretación se realiza en paralelo con la detección y captura de eventos, mientras que otras veces se hace después de recoger toda la información. En el primer caso (que presenta interesantes problemas estadísticos de estimación) se dice que se tiene un sistema de medida en tiempo real. Los sistemas de medida en tiempo real tienen la posibilidad de ser utilizados para el control dinámico de las prestaciones del sistema, mientras que el análisis diferido se usa para la planificación y ajuste del sistema a largo plazo.

2.4. Clasificación de los monitores

2.4.1. Introducción

La monitorización de un sistema informático no es una tarea habitual, por lo que estos sistemas no se diseñan (generalmente) para ser monitorizados, lo que suele añadir una complejidad adicional a la hora de diseñar un monitor. No obstante, en sistemas grandes y en muchos de los multiprocesadores que se diseñan actualmente, el monitor está integrado en el sistema y se diseña al mismo tiempo que el sistema informático.

El desarrollo de herramientas de medida para computadores, bajo una carga dada, ha seguido caminos diferentes, separados según las técnicas empleadas para su realización.

De este modo, los monitores se clasifican, según su implantación en:

— *Monitores software*, que son programas o ampliaciones del sistema operativo que acceden al estado del sistema, informando al usuario (ya sea de forma continua, o cuando éste lo requiere) sobre dicho estado.
— *Monitores hardware*, que son dispositivos electrónicos que se conectan a determinados puntos del sistema, donde se encargan de detectar determinados niveles o señales eléctricas que caracterizan la evolución del sistema.

— *Monitores híbridos*, que son una combinación de las dos técnicas anteriores, intentando combinar las ventajas de una y otra.

Según su mecanismo de activación se clasifican en:

— *Monitores de eventos o acontecimientos*, que son aquellos que se activan por la aparición de ciertos eventos. Si el evento se da con frecuencia, la sobrecarga que producen es elevada. Este evento puede ser también una decisión externa del responsable del sistema.
— *Monitores de muestreo*, que son aquellos que se activan a intervalos de tiempo fijos o aleatorios mediante interrupciones de reloj. La frecuencia de muestreo viene determinada por la frecuencia de aparición del estado que se desea analizar y por la resolución deseada.

Y según su forma de mostrar los resultados se clasifican en:

— *Monitores en tiempo real*, que constan de un módulo analizador que procesa los datos recogidos a medida que los recibe. En consecuencia, el módulo de proceso debe ejecutarse de forma continua, o con gran frecuencia, para poder presentar resultados parciales al analizador que informará de la evolución del sistema.
— *Monitores batch*, que difieren de los anteriores en que primero se recoge la totalidad de la información y, una vez terminado el período de recolección, se procesa dicha información.

Las tres clasificaciones pueden utilizarse de forma conjunta, así pues, un monitor puede clasificarse como software, conducido por muestreo y con análisis en *batch* de los resultados. La clasificación según el tipo de interfaz es la más habitual.

En la siguiente tabla se muestra cómo realizan cada una de las funciones los monitores software, hardware e híbrido.

Función	Monitor software	Monitor hardware	Monitor híbrido
Instrumentación	*Traps* Interceptación Muestreo	Sondas Conexión permanente	*Traps* Sondas
Selector	Programa	Programa Lógica	Programa Lógica
Procesador	Programa	Lógica Programa	Lógica Programa
Registrador	Memoria Disco	Contador Memoria Disco	Contador Memoria Disco
Analizador	Programa	Programa	Programa

Hay que señalar que, en la tabla precedente, mientras las referencias a programa de los monitores software son ejecutadas por la propia CPU del sistema que se está analizando, en los monitores hardware corresponden a una CPU distinta y en los híbridos pueden darse los dos casos.

2.4.2. Monitores software

Los monitores software son los más fáciles de encontrar en un sistema informático y aunque son utilizados de forma general para estudiar el comportamiento de un sistema son, sobre todo, adecuados para monitorizar los sistemas operativos, las redes y las bases de datos así como las aplicaciones que las utilizan.

Cada activación del monitor implica la ejecución de varias instrucciones por parte de la CPU del sistema que se está analizando, lo que puede provocar una gran sobrecarga en el sistema si la causa de la activación se produce con gran frecuencia.

Los monitores software pueden extraer información del sistema a través del seguimiento de los acontecimientos o eventos (mediante *traps* o interceptación) o muestrear los estados en que se halla. Las dos técnicas para la obtención de datos no son mutuamente excluyentes, sino que pueden ser utilizadas a la vez por una misma herramienta.

La aportación de código que necesitan los monitores software se puede realizar de tres formas distintas:

— Adición de un nuevo programa.
— Modificación del software que se va a medir.
— Modificación del sistema operativo.

El primer método es el más fácil de utilizar, a la vez que puede ser eliminado fácilmente, cuando ya no se necesite tomar medidas. Además se garantiza la integridad del sistema operativo y del propio programa que se va a medir. Generalmente las funciones del programa monitor se limitan a tomar medidas del sistema así como de los programas que se van a medir.

El segundo método está basado en el uso de sondas software, las cuales no son más que conjuntos de instrucciones insertadas en puntos clave del programa que debe observarse. Cada sonda debe detectar la llegada del flujo de control al punto crítico donde está situada, con lo que se puede determinar el número de veces y los instantes que se pasa por ese lugar, a la vez que el contenido de ciertas variables cuando se está ejecutando esa zona de código.

El tercer método se utiliza cuando determinadas variables que se deben medir sólo están disponibles para el sistema operativo, como pueden ser relojes de tiempo real, estado de colas del planificador, etc.

Como el monitor software necesita recursos del sistema, puesto que ocupa memoria, consume tiempo de CPU y realiza operaciones de E/S, se debe vigilar que se cumplan los siguientes requerimientos:

— La extracción de datos del sistema debe realizarse de forma cualitativa y cuantitativa.
— Las modificaciones del sistema operativo deben ser las mínimas posibles.
— La alteración de las características de carga del sistema, y, por lo tanto, de sus prestaciones no debe ser apreciable.
— Los recursos requeridos deben ser los mínimos posibles.

Existe un juego de intereses contrarios en cuanto a la precisión de las medidas y a la interferencia que éstas puedan ocasionar, ya que si se requiere una gran precisión de resultados, se tendrán que realizar un gran número de observaciones, lo que provocará una sobrecarga del sistema que hará que las medidas obtenidas no sean fiables.

Por otro lado, si no se quiere sobrecargar mucho el sistema y se realizan pocas observaciones, éstas pueden no recoger todo lo que realmente está sucediendo en el sistema, por tanto, no tendrán validez (o no serán significativas).

El monitor debe ofrecer la posibilidad de habilitar/deshabilitar, con facilidad, la monitorización, total o de determinadas partes, para evitar sobrecargar el sistema cuando no se utilice o no se utilicen determinadas partes del monitor.

En cuanto a su prioridad, si el monitor se ejecuta asíncronamente, la prioridad debe ser baja para no afectar a las operaciones importantes del sistema, pero si es necesario un registro minucioso del funcionamiento del sistema, su prioridad deberá ser alta. En general, su prioridad deberá ser superior a la de los acontecimientos que debe observar.

En cualquier caso, puesto que para su funcionamiento requieren de la ejecución de instrucciones del propio código del sistema analizado, habrá que tener presente en todo momento que teóricamente no pueden detectar acontecimientos que se produzcan por debajo del tiempo de ejecución de una instrucción ni acceder a acontecimientos que no se reflejen en alguna posición de memoria del sistema que sea accesible por la ejecución de una instrucción.

La mayoría de los monitores están escritos en lenguajes de programación de bajo nivel, tales como ensamblador o C, para mantener al mínimo la sobrecarga. Ya que un monitor software forma parte, generalmente, del sistema operativo, es mejor escribir ambos en el mismo lenguaje de programación.

A) Monitores software conducidos por acontecimientos

Existen dos tipos de instrumentación en los monitores conducidos por eventos o acontecimientos:

— Por *traps*, o segmentos de código que se insertan en determinadas partes del programa. Cuando el flujo de control llega a ellos, se ejecutan, realizando determinadas acciones encaminadas a señalar la aparición de ese evento y a extraer la información que lleva asociada.
— Por interceptación de las interrupciones, que desvían el tratamiento de la interrupción, redireccionando dicho tratamiento a una zona de código donde se halla el monitor, el cual contabiliza el tipo de interrupción y recopila la información asociada al acontecimiento que ha provocado la interrupción.

B) Monitores software de muestreo

Un monitor de muestreo interfiere mucho menos en el funcionamiento del sistema que un monitor conducido por eventos. Además, es posible variar sus dos principales características con el fin de ajustar la interferencia a unos niveles adecuados. Así, se puede variar la frecuencia de muestreo y el número de variables que hay que medir en cada uno de estos instantes de muestreo, teniendo en cuenta que la interferencia aumenta con el número de medidas que se toman y con la frecuencia elegida.

El procedimiento de extracción de datos utiliza un reloj, normalmente un temporizador interno, que provoca una interrupción del sistema a intervalos regulares de tiempo, para la extracción de los datos concernientes al estado del mismo.

Otras veces se procede a la recolección de datos a intervalos aleatorios aunque con una longitud media fija, para evitar que un posible sincronismo entre la evolución de los estados y los intervalos fijos impida al monitor detectar todos los estados del sistema. Otra solución para evitar el sincronismo se basa en elegir un período de muestreo que sea un submúltiplo del que utiliza el sistema operativo en el lanzamiento de aquellas funciones que se realizan a intervalos periódicos. La sincronización entre muestreo y eventos puede ser detectada fácilmente.

La figura 2.2 muestra el esquema funcional de un muestreador en un sistema conducido por interrupciones provocadas por una señal de reloj.

Lógicamente, el programa de extracción debe ser el de mayor prioridad, y debe ejecutarse con las interrupciones inhibidas. Por tanto, es necesario que el programa sea muy rápido (poca demanda de CPU), con el fin de no interferir en las prestaciones del sistema.

Los datos que se deben tomar son de dos tipos:

— Acerca de las actividades suspendidas cuando se provocó el disparo del temporizador.
— Acerca de las actividades en progreso, como el estado de canales, contenido de ciertas áreas de memoria, etc.

Los datos recolectados se deben almacenar de la forma más compacta posible. Generalmente, se subdividen en grupos temporales, formando lo que se denominan *registros lógicos*, que no son más que un conjunto de datos recogidos durante un período de tiempo.

Este período de tiempo puede ser de unos pocos minutos o, en el caso de tiempos de muestreo cortos, de segundos.

Si se supone que se desea conocer el estado de la CPU y del canal cada 333 ms, suponiendo un período de extracción de 8 horas (un día), se tendrá que el

número de muestras se eleva a 8 x 60 x 60 x (1000/333) = 86486 muestras. Considerando registros lógicos de 5 minutos, este número se ve reducido a 8 x 12 = 96 datos, conteniendo cada uno la media de 5 x 60 x (1000/333) = 900 muestras.

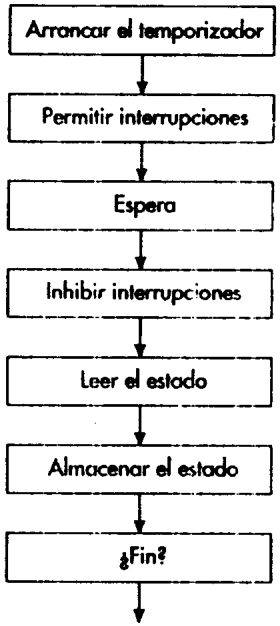

FIGURA 2.2. Esquema funcional de un muestreador en un sistema conducido por interrupción provocada por una señal de reloj.

a) Consideraciones temporales

Un monitor software debe ser capaz, entre otras cosas, de dar una duración de los fenómenos observados; es decir, de dar medidas temporales.

Esto lo puede realizar una herramienta software de dos formas: por muestreo o realizando medidas directamente. El primer método da medidas estadísticas en cuanto a tiempos medios de utilización de recursos, a partir de los cuales se pueden realizar medidas indirectas de otros fenómenos, como fallos de página, etc.

El método de tomar medidas directas es más interesante, por cuanto es capaz de dar la duración exacta de los fenómenos (instante de finalización menos instante de comienzo), así como la traza de acontecimientos del sistema, pero tiene el inconveniente de ser más difícil de implantar, sobre todo si se tiene en cuenta que algunos procesos entran con un alto grado de prioridad.

Ambos tipos de monitores software necesitan de un reloj en el sistema. Y en el caso de muestreadores, este reloj debe ser capaz, además, de realizar interrupciones en el sistema.

Una forma muy usual de realizar este reloj es utilizar una posición de la memoria principal o un registro interno del procesador como un contador binario que es incrementado (o decrementado) por un dispositivo hardware capaz de generar pulsos a intervalos regulares (generalmente un cristal de cuarzo). La frecuencia de este reloj da la resolución del sistema, es decir, la mínima cantidad de tiempo medible por él.

Muchas veces existen relojes paralelos que se inicializan cuando se arranca un proceso, salvando el estado de estos relojes cuando el proceso se suspende y recargándolo cuando éste se carga de nuevo. Cuando el proceso concluye, en este reloj se tiene el tiempo de ejecución del mismo. Lógicamente, el manejo de estos relojes corre a cargo del sistema operativo.

Existen algunas causas que hacen que este método no sea del todo exacto. Las principales son:

— Existe un retardo variable entre el comienzo y el final de la actividad y el arranque y la parada del reloj lógico asociado.
— El sistema operativo puede no ser totalmente honesto en cuanto a la adjudicación de los tiempos, y así el tiempo de tratamiento de determinadas interrupciones es cargado al proceso que tiene la CPU en ese momento.
— El acceso al reloj es normalmente una tarea del supervisor, y el tiempo de entrada de éste es aleatorio, cargando este tiempo al reloj lógico del proceso.

Otro elemento importante en la precisión de las medidas directas es la resolución del reloj. Si la resolución es demasiado pequeña (el tiempo entre dos ticks consecutivos es demasiado grande), es posible que el reloj no sea adecuado para la medida de la duración de determinadas actividades, o bien que los resultados no tengan la precisión necesaria. Se puede comprobar con un ejemplo en que se supone que hay s milisegundos entre dos ticks consecutivos (figura 2.3).

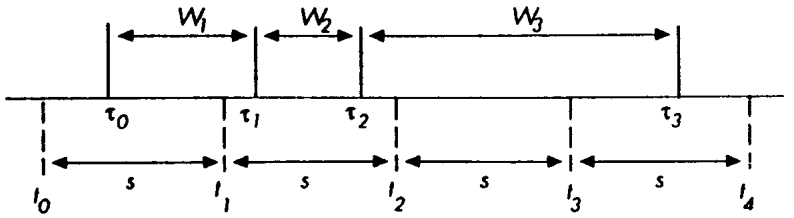

FIGURA 2.3.

W_1, cuya duración real es $T_1 = \tau_1 - \tau_0$ se considerará que tiene una longitud igual a $s = (t_1 - t_0)$, es decir, de menor longitud que la real.

W_2, cuya duración real es $T_2 = \tau_2 - \tau_1$ se considerará que tiene una longitud igual $0 = (t_1 - t_1)$, es decir, evidentemente de mayor longitud que la real.

W_3, cuya duración real es $T_3 = \tau_3 - \tau_2$ se considerará que tiene una longitud igual $2s = (t_3 - t_1)$, es decir, de mayor longitud que la real.

De manera general, se consulta el valor del reloj cuando comienza la actividad (instante T_i). Se supone que el valor real en ese instante es T_a, y que T es la duración del intervalo entre dos ticks del reloj (figura 2.4). Si la actividad termina en T_f y se consulta el reloj en ese momento, éste contendrá el valor T_b. La duración real de la actividad es de:

$$T_{real} = T_f - T_i$$

mientras que el valor medido es de:

$$T_{medido} = T_b - T_a$$

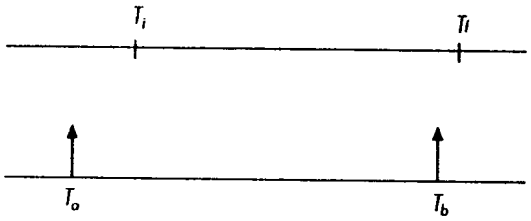

FIGURA 2.4.

Considerando el error como la diferencia entre lo que realmente dura y lo que se ha medido, se obtiene:

$$Error = T_{real} - T_{medido} = (T_f - T_i) - (T_b - T_a) = (T_f - T_b) - (T_i - T_a)$$

Tanto $(T_f - T_b)$ como $(T_i - T_a)$ pueden variar entre 0 y T, por lo que el error podrá variar entre $-T$ y T, o lo que es lo mismo

$$0 \leq |Error| \leq T$$

Es decir, que el error depende de T, por lo que la resolución debe de ser grande para que el error cometido en las medidas sea pequeño.

En caso de utilizar la técnica de muestreo existe un problema similar con el intervalo de muestreo, como muestra la figura 2.5.

Como puede verse, los resultados obtenidos serán los siguientes:

— A W_1 se le asigna 1 tick de reloj. Mucho más de lo que le corresponde.
— W_2 no se detecta.
— A W_3 se le asignan 2 ticks, único caso válido.

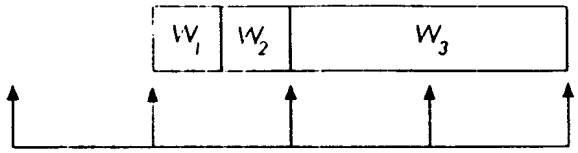

FIGURA 2.5.

Lo anterior se puede plantear de una manera más formal. De esta forma si se considera un muestreo genérico, como el de la figura 2.6, se tiene:

$$T_{real} = T_f - T_i$$
$$T_{estimado} = T_b - T_a$$

y por tanto:

$$Error = T_{real} - T_{estimado}$$
$$0 \leq |Error| \leq T$$

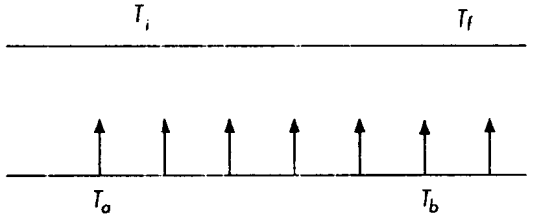

FIGURA 2.6.

Es decir, que cuanto mayor sea T, tanto mayor podrá ser el error, por lo que se deberán tomar intervalos de muestreo pequeños para disminuir el error. Lógicamente, se provocará una mayor sobrecarga tal y como se comentó anteriormente.

2.4.3. Monitores hardware

Son dispositivos para medir las presteciones de sistemas computadores y que se conectan al hardware del sistema que se va a monitorizar por medio de sondas electrónicas, que son elementos capaces de detectar eventos de tipo eléctrico (flancos, cambios de nivel, etc.). Con ellos se pueden deducir los índices de prestaciones más importantes, sobre todo, aquellos de más bajo nivel.

Su principal característica es que son externos al sistema que van a medir, lo que implica que:

— No utilizan recursos del sistema que se va a monitorizar.
— No producen interferencias con éste.
— Son muy rápidos.

Por tanto, en un principio parece que sean mejores que los software, pero tienen algunas desventajas frente a estos últimos. Las principales son:

— Son más difíciles de instalar.
— Existen magnitudes a las que sólo se puede acceder desde el software.
— Requieren, para su operación y para el análisis de sus resultados, de personal más especializado.
— Pueden interaccionar a nivel eléctrico con el sistema que se va a monitorizar, provocando perturbaciones que resulten en un funcionamiento anómalo del sistema monitorizado. No se deben utilizar nunca en medidas sobre sistemas críticos en funcionamiento real.

El esquema más simple de un monitor hardware está representado en la figura 2.7.

En ese monitor, los módulos lógicos tienen la misión de filtrar las señales, de forma que sólo se obtengan disparos cuando se produce una determinada combinación de las señales de entrada, lo que equivale a decir que se está produciendo un determinado evento.

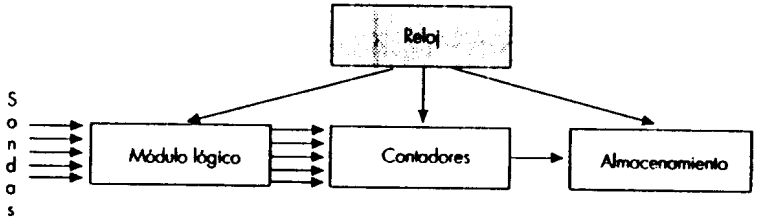

FIGURA 2.7. Esquema simple de un monitor hardware.

Los contadores se encargan de registrar el número de veces que se produce un determinado fenómeno. La parte de almacenamiento tiene la misión de memorizar de forma permanente el valor de los contadores. Puede ser opcional. El reloj se encarga de dar la señal de sincronismo con la que se deben captar los eventos.

Una realización muy simple del esquema anterior se puede ver en la figura 2.8. Se pretende determinar el tiempo de utilización de un determinado recurso. Este tiempo se caracteriza por un nivel alto en la salida de un biestable. Esta salida es interceptada por las sondas, las cuales llevan el nivel lógico de salida al módulo lógico, que en este sencillo caso no es más que una puerta AND cuya

otra entrada está conectada a la señal de reloj del monitor. De esta forma, la salida del módulo lógico será un tren de pulsos si la entrada está a nivel alto.

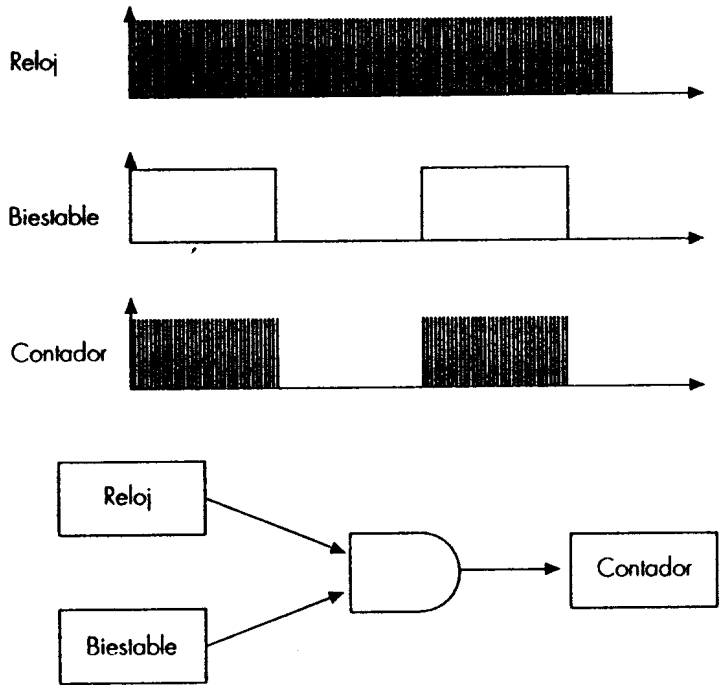

FIGURA 2.8.

La salida de este módulo lógico se lleva a un contador para que registre el número de pulsos recibidos. En otro contador se tiene directamente la señal de reloj.

La salida del monitor son las cuentas de los contadores, ya que la razón entre éstas da directamente el tanto por ciento de ocupación del dispositivo, que es lo que se pretendía conocer.

De este sistema tan simple se puede ir, por adición de módulos, complicando al monitor, con el fin de dotarle de más potencia.

El siguiente paso consiste en sustituir parte de la batería de contadores por una memoria RAM encargada de registrar de forma secuencial los eventos que se van produciendo, así como el instante de su aparición. En la figura 2.9 se puede ver un esquema que responde a estas características donde, de hecho, se han sustituido los distintos elementos por un microprocesador.