Análisis y planificación de sistemas distribuidos de tiempo real estricto

J. J. Gutiérrez García
gutierjj@ctr.unican.es
Tfno.: 942-201394
Fax: 942-201402

M. González Harbour
mgh@ctr.unican.es
Tfno.: 942-201483
Fax: 942-201402

Departamento de Electrónica y Computadores, Universidad de Cantabria
Avda. de los Castros s/n, 39005 Santander

Resumen: En este trabajo hacemos una revisión de la problemática vinculada al análisis y a la planificación de sistemas distribuidos de tiempo real estricto. Destacamos algunos de los trabajos más relevantes en este campo, así como algunas aportaciones que nosotros hemos realizado. Por un lado, detallamos los modelos utilizados para la representación de los sistemas sobre los que se puede efectuar el análisis de tiempo real, junto con algunas de las redes de comunicación que se utilizan en la interconexión entre los procesadores del sistema distribuido, y sobre las que se ha realizado una planificación para tiempo real. Por otro lado, hacemos referencia a las técnicas de análisis y a los algoritmos de planificación, aplicables a los sistemas distribuidos de tiempo real estricto que se pueden representar por los modelos descritos, y que utilizan una red de comunicación de tiempo real.

Palabras clave: Sistemas distribuidos, tiempo real, análisis, planificación, asignación de prioridades.

1. Introducción

La arquitectura típica de un sistema de control distribuido de tiempo real consta de varios nudos procesadores interconectados a través de una o más redes de comunicación. El software del sistema se compone de varias tareas concurrentes que normalmente están localizadas de forma estática en los nudos procesadores, ya que los efectos de una localización dinámica son muy difíciles de predecir para sistemas de tiempo real estricto. Normalmente, la comunicación entre las tareas del sistema (dentro del propio nudo o entre diferentes nudos) se realiza a través del intercambio de mensajes. También es normal que los mensajes intercambiados entre tareas de diferentes nudos se asignen de forma estática a las redes de comunicación. Además, es posible que las tareas de un mismo nudo compartan datos mediante los mecanismos de sincronización habitualmente utilizados en sistemas de memoria compartida. Dos aspectos importantes que se plantean en este tipo de sistemas son el análisis para determinar la planificabilidad de los *recursos* (procesadores y redes de comunicación), es decir, si se van a cumplir o no todos los requisitos temporales incluso en el peor caso posible, y la asignación de prioridades a las *acciones* (tareas en los procesadores y mensajes en las redes de comunicación) que optimice la posibilidad de alcanzar todos esos requisitos temporales.

El análisis de planificabilidad está bastante bien resuelto a partir de la aplicación de la teoría *RMA* (*Rate Monotonic Analysis*) [5] a los sistemas distribuidos. Para la aplicación de estas técnicas derivadas de la teoría RMA a sistemas distribuidos [8][14] se modela cada red como si fuera un procesador y cada mensaje como si fuera una tarea. Basándose en la técnica de análisis es posible realizar la asignación de prioridades a las tareas y a los mensajes en estos sistemas utilizando algoritmos de optimización que tratan de buscar la asignación óptima de manera que las diferentes tareas y mensajes cumplan los plazos que se les ha asignado [2][12]. En esta misma línea se trata de resolver incluso el problema de la asignación de acciones a recursos [12]. Como veremos más adelante, las técnicas de análisis deben tener en cuenta el efecto de la activación retrasada o *jitter*.

En este trabajo vamos a hacer una revisión de las técnicas de análisis y planificación aplicables a sistemas distribuidos de tiempo real estricto, destacando aquellos aspectos que nos van a permitir la automatización de dichas técnicas, y el desarrollo posterior de herramientas de análisis y asignación de prioridades. Así, en el apartado 2 detallamos los modelos de descripción para un sistema distribuido de tiempo real estricto. En el apartado 3 incidimos en la problemática existente en torno a las redes de comunicación de tiempo real desde el punto de vista de su planificación. El apartado 4 lo dedicamos a las técnicas de análisis y planificación basadas en RMA que se pueden aplicar sobre un sistema distribuido representado con los modelos descritos en el apartado 2. En el apartado 5 revisamos las técnicas de asignación de recursos y de prioridades que se apoyan en el análisis RMA. Finalmente, en el apartado 6 expresamos nuestras conclusiones.

2. Modelo de sistema distribuido

2.1. Modelo lineal

Consideramos un modelo lineal de sistema distribuido gobernado por eventos [2][4] en el que tenemos una serie de *eventos externos* (generados por dispositivos externos, temporizadores, etc), a los que responden un conjunto de tareas (distribuidas por los distintos procesadores) y de mensajes (distribuidos por las redes de comunicación), que se activan entre sí mediante la generación de *eventos internos*. Estos eventos internos los pueden producir tanto las tareas (para activar a otras tareas del propio procesador o a algunos mensajes de las redes de comunicación), como los mensajes (para activar a las tareas a las que van dirigidos). Suponemos que todas las secuencias de eventos externos que llegan al sistema se conocen por adelantado, y que también se conocen los ritmos con los que llegan estos eventos con requerimientos de tiempo real estricto. En otro caso, sería imposible garantizar la planificabilidad del sistema a priori, es decir, en tiempo de compilación. Los eventos internos se generan en las respuestas a los eventos externos, y por lo tanto, sus características se derivan de las de éstos. Suponemos además que los eventos son instantáneos por lo que no tienen influencia alguna en el análisis del sistema (los efectos de la sobrecarga se pueden incluir fácilmente en el análisis). También suponemos que las tareas se asignan estáticamente a los procesadores, y de igual manera los mensajes a las redes de comunicación. En muchos sistemas de tiempo real estricto las tareas están ligadas a procesadores específicos por la presencia de dispositivos *hardware* especiales que son necesarios para éstas.

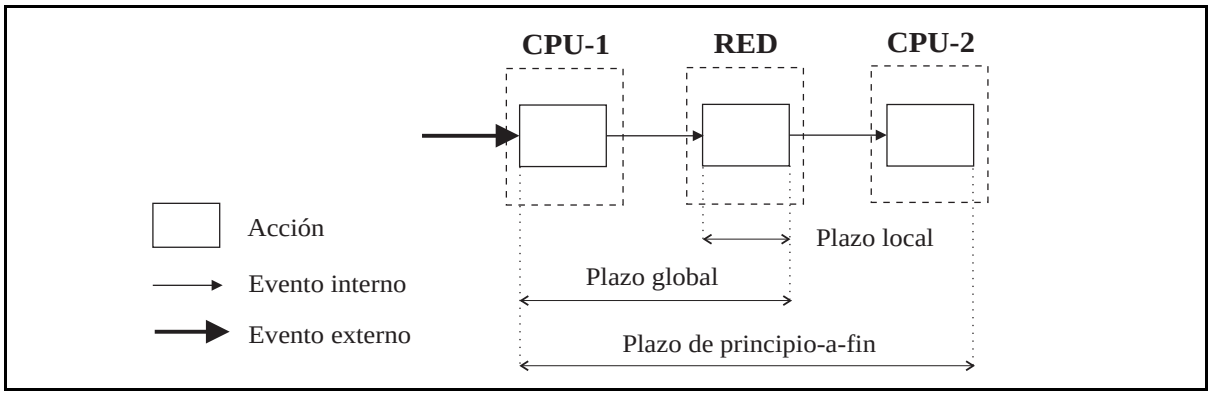

Figura 1. Respuesta lineal a un evento.

Cada evento externo que llega al sistema genera una respuesta en forma de una secuencia de acciones, que se activan entre sí a su vez mediante eventos internos. Cada una de estas acciones puede ser una tarea (o una parte de una tarea) ejecutada en un procesador, o un mensaje enviado a través de una red de comunicaciones. Normalmente, la primera acción de una respuesta es una tarea ejecutándose en algún procesador determinado. En este modelo de sistema distribuido las acciones solamente pueden ser activadas por un único evento (interno o externo), y solamente puede generar un evento interno activando por tanto a una única acción (en el mismo o en diferente recurso). La Figura 1 muestra la secuencia de respuesta a un evento externo junto con los requisitos temporales que se pueden imponer a las acciones lineales en su respuesta a los eventos externos, que son: *plazos globales* (relativos a la llegada del evento externo), *plazos locales* (relativos a la activación de cada acción en la respuesta), y *plazos de principio-a-fin* (plazos globales de la última acción en la secuencia de respuesta a un evento externo).

2.2. Modelo no lineal

El modelo lineal descrito no considera la posible sincronización entre tareas (salvo la exclusión mutua en el acceso a recursos compartidos), ni la activación de las mismas por combinaciones de eventos, y tampoco considera la posibilidad de generación de múltiples eventos por parte de una tarea. Para poder representar a un mayor número de sistemas de tiempo real que puedan presentarse en la práctica, se amplia el modelo lineal a un *modelo general* no lineal [4].

Al igual que para el modelo lineal, se define un modelo de sistema distribuido gobernado por eventos en el que se tiene una serie de eventos externos a los que responden un conjunto de tareas y de mensajes que

se activan entre sí mediante la generación de eventos internos. Sin embargo, en este modelo las características de los eventos internos (períodos y plazos) se pueden derivar de un solo evento externo o de una combinación de ellos.

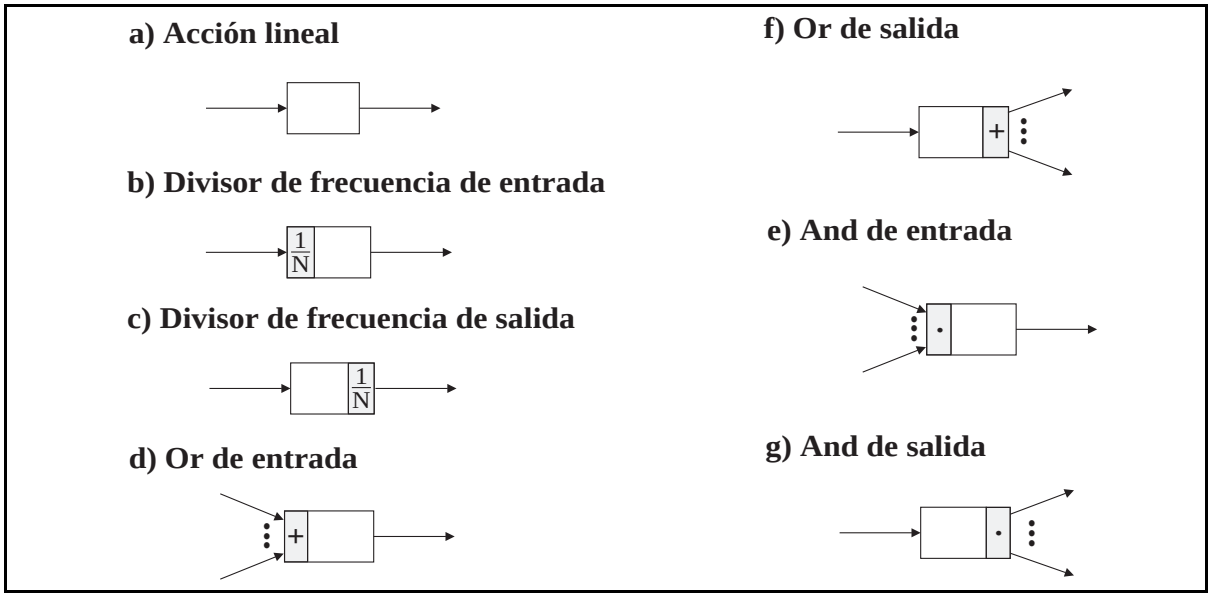

Figura 2. Modelo general de sistema distribuido gobernado por eventos.

La Figura 2 muestra los diferentes patrones a los que responden los eventos a la entrada y a la salida de las acciones, es decir, las diferentes combinaciones de eventos que pueden activar a una acción o las combinaciones de eventos que puede generar ésta. Una característica especial del patrón *and de entrada* es que todos los eventos de entrada deben tener idénticos períodos (no tiene sentido considerar que los eventos de entrada posean períodos diferentes, ya que esto supondría la acumulación de aquellos eventos más rápidos). Estos patrones reflejan un conjunto de combinaciones complejas de eventos que se pueden producir en sistemas reales, y que no se pueden describir con el modelo lineal.

Este modelo permite describir sistemas en los que aparezca cualquier combinación de acciones y eventos que respondan a los patrones de la Figura 2. Hay que considerar como excepción aquella combinación en la que un evento que es generado por una acción con *or de salida* da lugar a la activación de una acción con *and de entrada* en algún punto de su secuencia de respuesta [4].

3. Redes de comunicación de tiempo real

Aunque la mayoría de las redes de comunicación estándares no se adaptan bien a las comunicaciones de tiempo real, y no soportan una planificación de los mensajes basada en prioridades, existen algunas que se han utilizado para llevar a cabo comunicaciones de tiempo real estricto, tales como FDDI [1], token-ring [10], bus CAN [11], etc. También se han desarrollado algunas redes no estándares para hacer una planificación expulsora basada en prioridades.

La red de comunicación FDDI (Fiber Distributed Data Interface) es una red token-ring que no soporta un arbitrio de prioridades global que permita realizar directamente la planificación de actividades de tiempo real estricto. Sin embargo, Agrawal y otros [1] propusieron un método de planificación que garantizaba el cumplimiento de los plazos para mensajes síncronos y con una utilización de la red inferior al 33%. El método se basa en la utilización de un testigo temporizado que permite controlar el tiempo que el testigo tarda en rotar.

El IEEE 802.5 Token Ring es un estándar para redes de comunicación de 4 megabits por segundo desarrollado para aplicaciones comerciales, y que se puede configurar para soportar una planificación expulsora por prioridades. Aunque el campo reservado para la prioridad es de tres bits, se han desarrollado modelos y metodologías que extienden este campo de prioridad, y permiten utilizar técnicas avanzadas para la planificación en tiempo real [10].

Una red que soporta directamente la utilización de prioridades es el bus CAN (Controller Area Network). Los procesadores que se conectan a este bus se transmiten mensajes de tamaños entre 1 y 8 bytes. Los mensajes disponen de un único identificador representado por un número de 11 bits (en total 2032 identificadores, ya que CAN prohíbe el uso de identificadores con los 7 bits más significativos a 1). Este identificador sirve para asignar la prioridad al mensaje y para filtrar los mensajes en la recepción. Así, en el caso de que haya varios procesadores con mensajes que transmitir y el bus esté en silencio, todos intentan la transmisión a la vez, pero solamente al procesador con el mensaje de mayor prioridad se le permite transmitir, mientras que los demás deben retirarse hasta que el bus quede de nuevo en silencio. El protocolo utilizado en este arbitrio por prioridad evita las colisiones, y por lo tanto tiene un tiempo de respuesta totalmente predecible. El uso del identificador como prioridad del mensaje permite realizar una planificación de tiempo real basada en prioridades sobre este bus [11].

Por último, hacemos referencia a un trabajo anterior [4] en el que desarrollamos una herramienta software basada en la interfase de colas de mensajes definida en el POSIX.1b que permite el desarrollo de una comunicación con planificación mediante prioridades sobre subsistemas de comunicaciones estándares como el bus VME, y las líneas punto a punto. Esta herramienta permite el intercambio de mensajes entre tareas de manera transparente a los procesadores en los que se encuentren éstas (las tareas se comunican a través de colas de mensajes sin importar dónde estén localizadas). Además, se implementa el servidor esporádico para comunicaciones como mecanismo que permite eliminar el jitter [3], lo que permite como veremos más adelante incrementar la planificabilidad.

4. Técnicas de análisis y planificación

En un sistema distribuido hay múltiples recursos procesadores (CPUs) y uno o más recursos de comunicación (buses, redes LAN, líneas serie, etc). El problema de la planificación y del análisis de la respuesta temporal de programas concurrentes ejecutando en los recursos procesadores está bastante bien definido. Asumimos una planificación expulsora basada en prioridades fijas, porque está soportada directamente en sistemas operativos comerciales y lenguajes de programación estándares, y además permite utilizar la teoría RMA en el análisis del comportamiento temporal de peor caso. El análisis del tráfico de mensajes en estos recursos de comunicación se realiza utilizando las mismas técnicas RMA que para los recursos procesadores. Por lo tanto, se puede tratar a los mensajes en los recursos de comunicaciones de la misma manera que a las tareas en los recursos procesadores, excepto por un pequeño término de bloqueo que se debe tener en cuenta porque los mensajes están compuestos por secuencias de paquetes, cada uno de los cuales es indivisible y no expulsable. En algunas redes se deben considerar además otros parámetros en el análisis, como el tiempo de rotación del testigo en una red token-ring, o un término de bloqueo igual al tiempo de transmisión del mensaje de mayor longitud si la planificación de la red no contempla la división del mensaje en paquetes.

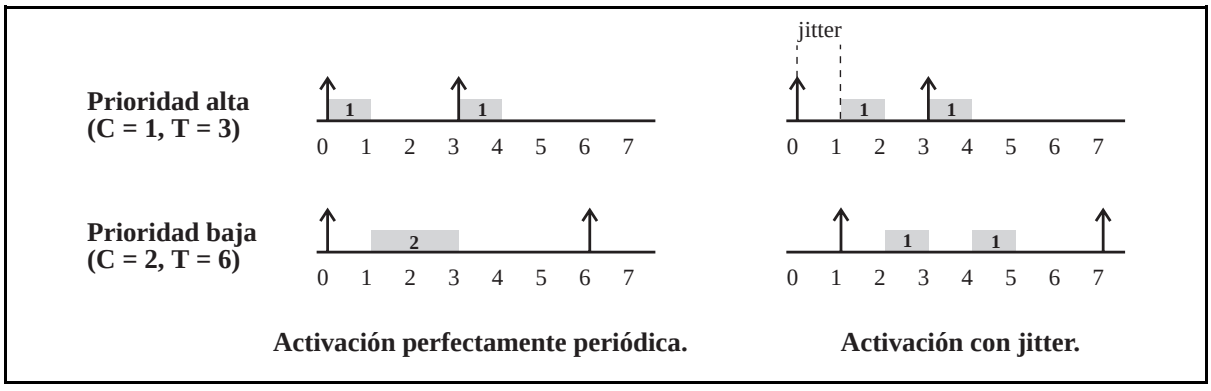

Figura 3. Efectos del jitter sobre una acción de baja prioridad.

El análisis de un sistema distribuido descrito por los modelos del apartado anterior se puede llevar a cabo mediante las técnicas basadas en RMA, pero teniendo en cuenta el efecto del jitter [7][14]. La Figura 3 muestra un ejemplo sencillo en el que una acción de prioridad baja únicamente puede ser expulsada una vez por otra acción con mayor prioridad, para el caso en el que ésta se active de manera perfectamente periódica. Sin

embargo, esta acción de prioridad baja puede llegar a ser expulsada dos veces si la activación de la de prioridad alta se retrasa, es decir, sufre el efecto del jitter (el instante de activación no coincide con el comienzo de la ejecución). En los sistemas distribuidos los mensajes se activan a partir de la ejecución de las tareas, y sus instantes de activación no son perfectamente periódicos porque dependen de los instantes en los que finalizan las tareas que los disparan (que son variables). Exactamente lo mismo ocurre con las tareas al ser activadas por la llegada de un mensaje. Existen algoritmos de planificación como el servidor esporádico capaces de eliminar el jitter, que son aplicables tanto en los procesadores [9] como en las redes de comunicación [3]. La eliminación del jitter puede suponer un incremento en la utilización de los recursos del sistema de hasta un 50% [3].

El análisis de una acción en un recurso se realiza a partir del cálculo de la longitud del período de ocupación de peor caso, para obtener después la respuesta de peor caso. Así, el período de ocupación para la tarea τ_i se calcula mediante la siguiente ecuación utilizada de manera iterativa [6][13]:

$$w_i^{n+1}(q) = (q+1)\ C_i + \sum_{\forall j \in hp(i)} \left\lceil \frac{w_i^n(q)}{T_j} \right\rceil C_j$$

donde, *hp(i)* es el conjunto de tareas que pueden expulsar a la tarea τ_i, o sea, que tienen mayor prioridad, C_i es el tiempo de ejecución de peor caso de la tarea τ_i, T_j es el período de la tarea τ_j, $\lceil x \rceil$ es la función techo sobre x (redondeo de x al entero superior), y $w_i(q)$ es el tiempo de finalización de peor caso de la tarea τ_i para la *q-ésima* activación (después de un *instante crítico*, en el que todas las tareas se activan a la vez). La ecuación se aplica iterativamente para sucesivos valores de n, partiendo de un valor inicial $w_i^0(q) = (q+1)\ C_i$. El valor de $w_i(q)$ viene determinado por el valor al que converge la ecuación cuando $w_i^{n+1}(q) = w_i^n(q)$.

Esta ecuación se aplica de forma sucesiva para valores de q iguales a 0,1,2,…, y termina cuando se verifica la desigualdad $w_i(q) \le (q+1)\ T_i$. En ese momento se ha finalizado todo el trabajo de prioridad P_i o superior, y por consiguiente, se ha obtenido la longitud del período de ocupación de peor caso, igual a $w_i(q)$. A partir de los resultados obtenidos para los distintos valores de q, se puede calcular la respuesta de peor caso r_i para la tarea τ_i restando del tiempo de finalización de la activación *q-ésima*, el instante de activación correspondiente $r_i = \max_{q=0,1,2,...} (w_i(q) - qT_i)$.

El efecto del jitter es necesario tenerlo en cuenta a la hora de calcular los tiempos de respuesta de peor caso. De este modo, se modifica la ecuación mediante la que se calcula el período de ocupación, para incorporar el jitter al efecto que las tareas τ_j de mayor prioridad tienen sobre la tarea τ_i, obteniéndose la siguiente expresión [13]:

$$w_i^{n+1}(q) = (q+1)\ C_i + B_i + \sum_{\forall j \in hp(i)} \left\lceil \frac{J_j + w_i^n(q)}{T_j} \right\rceil C_j$$

donde, J_j es el jitter correspondiente a la tarea τ_j y B_i es el término de bloqueo correspondiente a la tarea τ_i (mayor sección crítica de un semáforo cuyo techo de prioridad es mayor o igual que la prioridad de la tarea τ_i).

La técnica desarrollada por Tindell y Clark [14], aplicable a sistemas distribuidos que responden al modelo lineal, consiste en repetir el análisis de peor caso de forma iterativa (cálculo de respuestas de peor caso y jitter para todos los recursos del sistema). En cada iteración el jitter de una acción es igual a la respuesta de peor caso de la acción previa en la secuencia de respuesta al evento. Las características de monotonicidad del tiempo de respuesta con el jitter hacen que el análisis converja a una solución correcta. Como resultado del análisis se obtienen las *respuestas globales* de peor caso (desde que llega el evento externo hasta que finaliza la ejecución de la acción, que para la acción a_i se expresa como $R_i = \max_{q=0,1,2...} [w_i(q) - qT_i + J_i]$), y a partir de ellas, se puede comprobar que se verifican las condiciones de planificabilidad impuestas al sistema distribuido. El sistema es planificable cuando se verifican los plazos globales asignados a las acciones, o sea, cuando las respuestas globales de peor caso de las acciones son menores o iguales que sus plazos globales. Si el sistema tiene impuestos plazos locales, es necesario además verificar su cumplimiento mediante la comparación con las *respuestas locales* de peor caso (desde que se activa una acción hasta que finaliza su ejecución). En [7] se

obtiene una estimación para estas respuestas locales de peor caso, de manera que se puede extender la caracterización de un sistema como planificable con la verificación de los plazos locales.

Esta misma técnica de análisis se puede extender para su aplicación al modelo general [4]. Para ello, se transforma el sistema de tareas y mensajes en un sistema equivalente sobre el que se pueden aplicar las ecuaciones desarrolladas para el modelo lineal. Este sistema equivalente establece los parámetros para el análisis correspondientes a las acciones (períodos y plazos) a partir de la descripción del sistema realizada mediante el modelo general (características de los eventos del sistema y de sus respuestas), y también determina las transformaciones que se deben realizar y el modo en el que se debe aplicar el análisis sobre algunos patrones de eventos que no son analizables directamente mediante la técnica aplicable al modelo lineal. A grandes rasgos el sistema equivalente pretende linealizar los patrones de llegada y generación de eventos definidos en el modelo general, para que sean analizables.

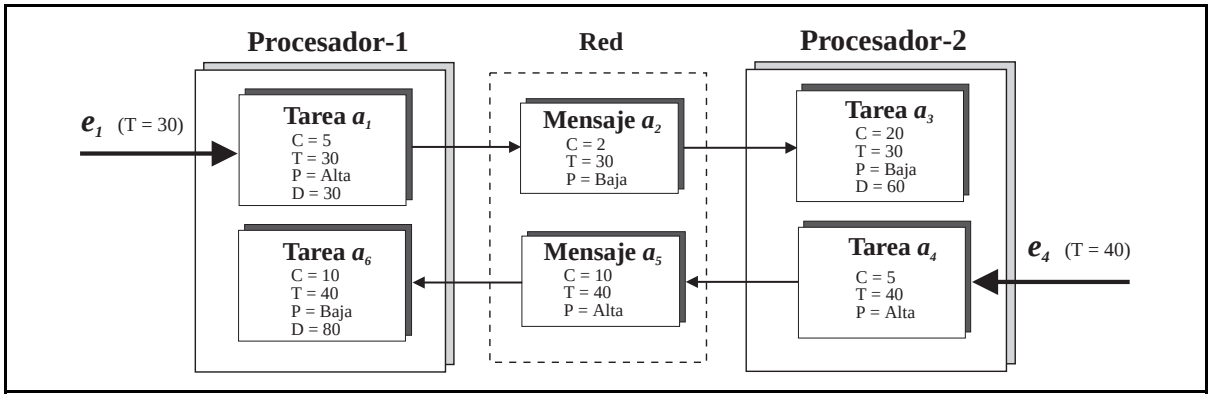

Figura 4. Ejemplo de sistema distribuido de tiempo real estricto.

A continuación, introducimos un ejemplo sencillo de sistema distribuido con requerimientos de tiempo real en el que vamos a resaltar los aspectos más importantes del modelado, de la aplicación de la técnica de análisis sobre el modelo, y de los resultados que se obtienen. En la Figura 4 se muestra un sistema distribuido de tiempo real que responde al modelo lineal. El sistema se compone de dos procesadores conectados a través de una red de comunicación, y debe responder a dos eventos externos que son periódicos. Cada respuesta a uno de estos eventos está formada por una secuencia de acciones que está distribuida a lo largo del sistema (una tarea responde al evento externo en un procesador y envía un mensaje a través de la red a otra tarea en el otro procesador).

Desde el punto de vista del modelado, las acciones estás caracterizadas por su período *T* (el mismo que el de su evento externo), su tiempo de ejecución de peor caso *C* (o tiempo de transmisión de peor caso para los mensajes), su prioridad *P* (para simplificar *Alta* o *Baja*), y los plazos globales *D* que se les haya impuesto. No se consideran plazos locales para simplificar el ejemplo, teniéndose las tareas a_3 y a_6 con plazos globales de principio-a-fin, y la tarea a_1 con un plazo global. La Tabla I muestra los resultados de la aplicación del análisis temporal al ejemplo, es decir, los valores a los que convergen los términos de jitter y las respuestas globales de peor caso para cada una de las acciones. A partir de los resultados del análisis se puede comprobar si se verifican los plazos, y por lo tanto, si el sistema es planificable (el sistema del ejemplo es planificable). Un último aspecto que se puede resaltar es que efectivamente las únicas acciones que se activan periódicamente son las tareas que responden a los eventos externos, ya que las demás presentan jitter.

Tabla I. Análisis temporal del ejemplo.

Acción	J_i	R_i
a_1	0	5
a_2	5	17
a_3	17	42
a_4	0	5
a_5	5	15
a_6	15	30

5. Técnicas de asignación de recursos y de prioridades

Las técnicas de asignación de prioridades parten de un sistema en el que se tiene un conjunto de tareas distribuidas en un conjunto de procesadores que se envían mensajes por las diferentes redes de interconexión, y tratan de asignar prioridades a las tareas y a los mensajes, de manera que se verifiquen los requerimientos de tiempo real impuestos. Tindell, Burns y Wellings [12] aplicaron la técnica del *templado simulado* (simulated annealing) a la resolución del problema de asignación de prioridades a tareas y a mensajes conjuntamente con la asignación de las tareas a los procesadores. Utilizaron una arquitectura distribuida muy específica constituida por un conjunto de procesadores que se comunicaban a través de un bus mediante un protocolo de paso de testigo, por lo que los mensajes estaban estáticamente asignados a la red de comunicación.

En [2] desarrollamos un nuevo algoritmo de planificación heurístico *HOPA* (Heuristic Optimized Priority Assignment) que resuelve el problema de la asignación de prioridades basándose en el conocimiento de los tiempos de respuesta del sistema. Este algoritmo se fundamenta en la distribución de los plazos globales de principio-a-fin de cada respuesta a un evento externo entre las diferentes acciones que componen esa respuesta. Una vez que se le ha asignado a cada acción un plazo local artificial, se asignan prioridades *deadline monotonic* (mayor prioridad para las acciones con plazos más pequeños) en cada recurso de acuerdo con estos plazos locales, y se lleva a cabo el análisis del sistema completo. Como resultado del análisis se calculan nuevos plazos intermedios. La iteración se ejecuta hasta que se encuentra una solución adecuada o se cumple algún criterio de parada.

Los algoritmos de planificación se apoyan en el análisis continuado del sistema, por lo que se puede pensar en el desarrollo de herramientas de planificación para sistemas distribuidos de tiempo real, que como caso particular realicen el análisis del sistema. Por otra parte, en algunas ocasiones puede resultar interesante que determinadas acciones del sistema tengan un orden relativo en sus prioridades. Para resolver estos casos introducimos el concepto de *prioridad preasignada* que es una prioridad dada a priori a una acción del sistema relativa a otras acciones con prioridad preasignada dentro de un recurso. Las herramientas de planificación tratarán de asignar prioridades a las acciones conservando el orden relativo a las prioridades preasignadas en cada recurso del sistema. En general, los algoritmos de planificación realizan básicamente las siguientes actividades:

```
algoritmo planificación is
begin
        Asignación inicial de prioridades;
        loop
                Cálculo de los tiempos de respuesta de peor caso;
                exit when criterio de parada;
                Nueva asignación de prioridades;
        end loop;
end planificación;
```

Para determinar la bondad de las soluciones alcanzadas por los algoritmos de planificación (tanto si el sistema es planificable como si no lo es), se utiliza el *índice de planificación* que se define en función de la peor distancia entre los tiempos de respuesta de peor caso y los plazos impuestos para cada acción [4].

6. Conclusiones

En este artículo hemos abordado el análisis y la planificación de sistemas distribuidos de tiempo real estricto poniendo de manifiesto los problemas que plantea la distribución, y las soluciones que se dan para resolver estos problemas. Las técnicas de análisis y de asignación de prioridades que se emplean derivan de la teoría RMA, y son aplicables a sistemas distribuidos que son representables mediante los modelos descritos. Hemos visto además cómo uno de los principales problemas que afectan a la planificabilidad de un sistema distribuido es el jitter, que se puede eliminar mediante el uso de algoritmos de planificación como el servidor esporádico. El empleo de estos algoritmos hace posible la simplificación del análisis, y por lo tanto, también de las técnicas de asignación de prioridades que se apoyan en este análisis.

Bibliografía

[1] G. Agrawal, B. Chen, W. Zhao, y S. Davari: "Architecture Impact of FDDI Network on Scheduling Hard Real-Time Traffic". Workshop on Architectural Aspects of Real-Time Systems, Diciembre 1991.

[2] J.J. Gutiérrez García, y M. González Harbour: "Optimized Priority Assignment for Tasks and Messages in Distributed Hard Real-Time Systems". Proceedings of the 3rd Workshop on Parallel and Distributed Real-Time Systems, Santa Barbara, CA, pp. 124-132, Abril 1995.

[3] J.J. Gutiérrez García, y M. González Harbour: "Increasing Schedulability in Distributed Hard Real-Time Systems". Proceedings of the 7th Euromicro Workshop on Real-Time Systems, Odense, Denmark, pp. 99-106, Junio 1995.

[4] J.J. Gutiérrez García, y M. González Harbour (Director de la Tesis): "Planificación, Análisis y Optimización de Sistemas Distribuidos de Tiempo Real Estricto". Tesis Doctoral, Universidad de Cantabria, Octubre 1995.

[5] M. Klein, T. Ralya, B. Pollak, R. Obenza, y M. González Harbour: "A Practitioner's Handbook for Real-Time Systems Analysis". Kluwer Academic Pub., 1993.

[6] J.P. Lehoczky: "Fixed Priority Scheduling of Periodic Task Sets with Arbitrary Deadlines". IEEE Real-Time Systems Symposium, 1990.

[7] J.C. Palencia Gutiérrez, J.J. Gutiérrez García, y M. González Harbour: "On the Schedulability Analysis of Distributed Hard Real-Time Systems". Proceedings of the 9th Euromicro Workshop on Real-Time Systems, Toledo, pp. 136-143, Junio 1997.

[8] L. Sha, y S.S. Sathaye: "A Systematic Approach to Designing Distributed Real-Time Systems". IEEE Computer, pp. 68-78, Septiembre 1993.

[9] B. Sprunt, L. Sha, y J.P. Lehoczky: "Aperiodic Task Scheduling for Hard Real-Time Systems". The Journal of Real-Time Systems, Vol. 1, pp. 27-60, 1989.

[10] J.K. Strosnider, T. Marchok, y J.P. Lehoczky: "Advanced real-time scheduling using the IEEE 802.5 Token Ring". Proceedings of the IEEE Real-Time Systems Symposium, Huntsville, Alabama, USA, pp. 42-52, 1988.

[11] K. Tindell, A. Burns, y A.J. Wellings: "Calculating Controller Area Network (CAN) Message Response Times". Proceedings of the 1994 IFAC Workshop on Distributed Computer Control Systems (DCCS), Toledo, Septiembre 1994.

[12] K. Tindell, A. Burns, y A.J. Wellings: "Allocating Real-Time Tasks. An NP-Hard Problem Made Easy". Real-Time Systems Journal, Vol. 4, No. 2, pp. 145-166, Mayo 1992.

[13] K. Tindell: "An Extendible Approach for Analysing Fixed Priority Hard Real-Time Tasks". Journal of Real-Time Systems, Vol. 6, No. 2, March 1994.

[14] K. Tindell, y J. Clark: "Holistic Schedulability Analysis for Distributed Hard Real-Time Systems". Microprocessing & Microprogramming, Vol. 50, Nos.2-3, pp. 117-134, Abril 1994.