Revista Iberoamericana de Automática e Informática Industrial

Revista Iberoamericana de Automática e Informática Industrial 18 (2021) 288-299

www.revista-riai.org

Controlador difuso para compensar cargas de comunicación en sistemas en tiempo real

Aparicio-Santos, J.[a,*], Hermosillo-Gómez, J.[a], Benítez-Pérez, H.[a], Alvarez-Icaza, L.[b]

[a] *Instituto de Investigaciones em Matemáticas Aplicadas y en Sistemas, Universidad Nacional Autónoma de México; Coyoacán, 04510, Ciudad de México*
[b] *Instituto de Ingeniería, Universidad Nacional Autónoma de México; Coyoacán, 04510, Ciudad de México*

To cite this article: Aparicio-Santos, J., Hermosillo-Gómez, J., Benitez-Pérez, H., Alvarez-Icaza, L. 2021. Fuzzy controller to compensate comunication loads in real-time. Revista Iberoamericana de Automática e Informática Industrial 18, 288-299. https://doi.org/10.4995/riai.2021.14544

Resumen

Se presenta un administrador de recursos (*RM*) difuso para compensar las cargas de comunicación en sistemas en tiempo real. El diseño del *RM* se basa en un nuevo modelo de Servidor de Ancho de Banda Constante (*CBS*) que se encarga, a través de una plataforma virtual, de asignar tiempo de proceso a las tareas de mayor prioridad cuando existe capacidad disponible. Si se asume que cada aplicación puede ser ejecutada con diferentes niveles de servicio sin que este esté por debajo de un límite mínimo, se propone una aproximación difusa que permite ajustar los tiempos de proceso asignados a cada tarea. Esta aproximación permite compensar el comportamiento no lineal en las solicitudes de tiempo de proceso. El *RM* aumenta o disminuye la plataforma virtual para cada aplicación y le asigna un presupuesto máximo de tiempo de proceso, mismo que la aplicación usa gradualmente y que se reasigna al agotarse, sin por ello afectar el desempeño del resto de las aplicaciones. El esquema se auto-ajusta cuando ocurren a cambios repentinos en los requerimientos de tiempo de proceso de las aplicaciones.

Palabras clave: Manejador de recursos, servidor de ancho de banda constante, asignación de tiempo de procesador, control difuso, sistemas en tiempo Real

Fuzzy controller to compensate comunication loads in real-time

Abstract

A Fuzzy Resource Manager (*RM*) to compensate communication loads in real-time systems is presented. The design is based on a new model of a Constant Bandwidth Server (*CBS*), which is responsible for assigning time slots to tasks with the highest priority when idle time is available. Assuming that each application can be executed at different service levels, without being below a minimum limit, a fuzzy approach is introduced that allows to adjust the time resources assigned to each task and to compensate non-linearities in time resources requests. The *RM* increases or decreases the virtual platform for each application and assigns a maximum process time budget for it, which is gradually used and refilled when depleted, without affecting the other applications. The scheme self-adjusts to sudden changes in applications process time requirements.

Keywords: Resource manager, constant bandwidth server, timer resources allocation, fuzzy control, real-time systems.

1. Introducción

Un sistema en tiempo real (real-time system - *RTS*) está constituido por un conjunto de aplicaciones en tiempo real (*real-time applications* - *RTA*), que usualmente contienen tareas (tasks) que cooperan entre sí para lograr los objetivos del sistema. Estas tareas deben ser atendidas en intervalos de tiempos específicos, relacionados con la naturaleza del *RTS* (Clark, 1990), por lo que el correcto funcionamiento del *RTS* depende de no sólo de una toma de decisiones apropiada, sino también del momento en que se comunican los resultados (Stankovic, 1988).

*Autor para correspondencia: ja.aparicio.santos@gmail.com

Para lograr un buen desempeño del *RTS*, se plantea utilizar un administrador de recursos (*Resource Manager - RM*) difuso y distribuido. Lo primero, hace que el *RM* pueda lidiar con cambios no lineales en las necesidades de las tareas sin perder funcionalidad. Lo segundo, permite dividir la solución del problema. Así, por un lado, cada aplicación hace una regulación local de sus recursos, mientras que, por el otro, el *RM* global coordina la distribución de tiempo de procesamiento asignado a cada aplicación. Por lo anterior, es claro que el *RM* juega un papel crucial pues coadyuva a equilibrar el uso de los recursos entre las distintas aplicaciones y tareas de un *RTS*. El *RM* que se presenta en este trabajo usa un Servidor de Ancho de Banda Constante, *CBS*, cuya dinámica está descrita por ecuaciones en diferencias. Al combinar herramientas de la teoría control con algoritmos de planificación de recursos, se obtiene un esquema que hace posible analizar y probar la convergencia a una distribución justa (*fair allocation)* de los recursos entre las aplicaciones de un *RTS*, a partir una distribución inicial arbitraria.

1.1. Antecedentes

Uno de los primeros pasos en el diseño de un *RM* es definir una abstracción que represente los recursos a distribuir. Una manera de hacerlo es usar el término de calidad de servicio *(quality of service, QoS*) (Ganz et al., 2003), el cual, tiene múltiples significados y perspectivas, que dependen específcamente de cada aplicación. Así, el *RM* debe administrar la tasa de ejecución de las aplicaciones sin que estas pierdan *QoS*.

Para diseñar el *RM*, en Nesbit et al. (2008) se utiliza el esquema de máquina virtual privada, la cual está constituida por un conjunto de recursos de *hardware* virtuales, mientras que en Mok and Feng (2001) se le modela a través de un procesador virtual que divide a las tareas en grupos que comparten recursos, pero que no interfieren entre ellos; este procesador virtual está descrito por una proporción de uso y un retardo. Derivado de este modelo, Bini et al. (2011) y Chasparis et al. (2016) usan el concepto de plataforma virtual, la cual es una abstracción de la proporción de uso del procesador. En las técnicas de distribución de recursos, cada reservación de estos es vista como una plataforma virtual ejecutada en una fracción de la disponibilidad del procesador físico.

La plataforma virtual es manejada por un proceso llamado servidor (*server*), dedicado al manejo de los recursos compartidos. Un solo servidor puede manejar diversos subconjuntos de tareas, y si este fuera el caso, las tareas a cargo del servidor compartirán recursos, pero deben estar protegidas contra sobrecostes (*overrun*) (Buttazzo, 2011). Si una tarea es manejada por un solo servidor, entonces esta tarea está aislada temporalmente y se debe garantizar que los recursos que recibe no interfieran con los necesarios para realizar las demás tareas. En este trabajo se asume que una aplicación puede ser ejecutada a diferentes niveles de servicio (*service levels*), donde mayor nivel de servicio implica mayor QoS, (Chasparis et al., 2016).

El *RM* se encarga de distribuir los recursos, tratando de cumplir con la calidad de servicio requerida. Para elegir un esquema de *RM* se debe tomar en cuenta la información disponible. Los esquemas generales de *RM* son tres: centralizados, distribuidos e híbridos (Byeong Gi et al., 2009).

Un esquema clásico para diseñar *RM* es el esquema centralizado (Bini et al., 2011), donde el *RM* se encarga de asignar los procesadores virtuales a las aplicaciones, observa el uso de recursos y asigna el nivel de servicio de cada aplicación (Chasparis et al., 2013). Estos esquemas tienen la ventaja de ser muy estables, sin embargo, tienen varias debilidades. La primera de ellas es que la complejidad de los algoritmos usados para implementar el *RM* crece geométricamente con el número de aplicaciones, lo cual puede implicar que el manejo del *RM* llegue a consumir más recursos que los que debe distribuir. La segunda es que resulta difícil encontrar una función de costos para distribuir los recursos, pues el *RM* debe comparar la calidad de servicio de diferentes aplicaciones y, como se mencionó, la noción de calidad de servicio depende de cada una de estas y no es necesariamente la misma entre aplicaciones. Por último, para una apropiada asignación de los niveles de servicio el *RM* debe tener conocimiento de los estados internos de todas las aplicaciones; para ello, estas deben informar al *RM* sobre el nivel de servicio disponible y los recursos que se consumirán por cada nivel de servicio, lo que incrementa significativamente el costo de la comunicación (Chasparis et al., 2016).

Debido a dichas debilidades, los esquemas de *RM* distribuidos han llamado la atención de manera importante (Subrata et al., 2008). En un esquema distribuido, cada aplicación determina la cantidad y la utilización de los recursos por si misma, basándose en la información local. Un esquema distribuido permite la resolución del problema de la escalabilidad y la administración de los recursos con un pequeño intercambio de información. La principal distinción del esquema distribuido es que existen múltiples *RMs* en el sistema. La asignación óptima de recursos de una aplicación depende del funcionamiento de las otras aplicaciones. Como resultado, el esquema distribuido tiene como punto débil que la asignación global de recursos puede ser inestable, ya que la asignación de recursos de cada aplicación no converge necesariamente a un estado estable. Incluso si converge, el punto de equilibrio puede ser sólo una optimización local, y la optimización global no está garantizada (Byeong Gi et al., 2009).

El esquema híbrido, puede aprovechar las ventajas de los esquemas centralizados y distribuidos, el *RM* central no determina la asignación de recursos específicos de cada aplicación, sino que sólo distribuye alguna información de referencia útil para un funcionamiento estable y eficiente. Cada aplicación entonces determina el uso de los recursos basados tanto en la referencia distribuida y en su propia información local. El problema de la gestión de los recursos se divide en dos subproblemas: uno es la determinación de la información en el administrador central de recursos, y la otra es la determinación de la asignación de recursos locales en cada aplicación (Byeong Gi et al., 2009).

El uso de lógica difusa para la asignación de recursos computacionales se discute en Litoiu and Tadei (2001), que presenta modelos de planificación para tareas periódicas generales con *deadlines*, cálculo difuso de tiempos de procesamiento y algoritmos basados en la asignación óptima de prioridades. Los autores muestras que el uso de lógica difusa logra una mejora en el algoritmo *rate monotonic*, incluso con tareas muy restringidas.

El esquema de la Figura 1 muestra el *RM* híbrido que se utiliza en este trabajo, el cual asigna la proporción de uso del procesador, mientras cada aplicación regula su propio nivel de servicio. En este esquema se deben proveer condiciones para

obtener convergencia a distribuciones equitativas (*fair allocations*). El esquema presenta las siguientes características:

• Su complejidad crece linealmente con el número de aplicaciones.

• No se tiene que conocer los estados internos de cada aplicación.

• Es robusto frente al número y naturaleza de las aplicaciones (IEEE, 1990).

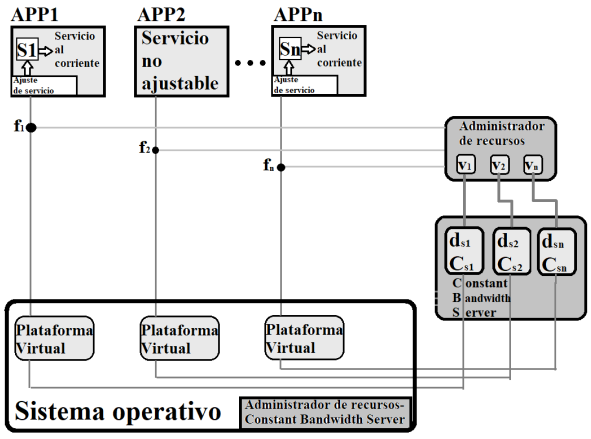

Figura 1: Esquema de *RM* híbrido usado en este trabajo, extensión del modelo presentado en (Chasparis et al., 2016) al que se incorpora un servidor de ancho de banda constante. Aquí *APPn* es la aplicación n, f_n la función de emparejamiento entre la proporción de uso v_n y el nivel de servicio s_n de la aplicación n (ver Ecuación (4)).

Se asume que cada *RTA* se compone de tareas que pueden ser síncronas o asíncronas y que a su vez están constituidas por porciones de código sensibles al tiempo llamadas *jobs*, donde los *jobs* que integran una tarea pueden repetirse (Chasparis et al., 2013). En este trabajo se considera que la *RTA* sólo ejecuta una tarea integrada por *jobs* que requieren atención en diferentes intervalos de tiempo. Se considera que existen aplicaciones *hard*, como la *APP2* en la Figura 1, que no admiten ajuste en su tiempo de ejecución y aplicaciones *soft*, como las *APP1* y *APPn* que sí pueden ajustarse y cuyo tratamiento es modificado por el *RM*.

El *RM* presentado hereda las siguientes propiedades del esquema presentado en Chasparis et al. (2016): evita inanición (*starvation avoidance*), obtiene balanceo (*balance*), converge síncrona y asíncronamente (*asynchronous and synchronous convergence*) (Chasparis et al., 2013).

La modificación más importante realizada en este trabajo al diseño presentado en Chasparis et al. (2016) es la incorporación de un servidor de ancho de banda constante *CBS* en el funcionamiento del *RM* híbrido que se muestra en la Figura 1. El *CBS* (Abeni and Buttazzo, 1998) es un mecanismo de servicio con el cual se implementa una estrategia para reservar recursos de proceso. Tanto el *RM* como el *CBS* están descritos por un conjunto de ecuaciones en diferencias, lo que facilita diseñar esquemas de control que garanticen la convergencia a distribuciones equitativas (*fair allocations*) y la estabilidad global para cualquier distribución inicial de recursos (por ejemplo: tiempo de procesamiento, ancho de banda de memoria, cache, etc.). El uso del nuevo *RM* se ejemplifica a través del sistema de control de un dispositivo electromecánico.

2. Administrador de recursos (*RM*)

En el *RM* de este trabajo, los recursos a distribuir son el tiempo de procesamiento (recurso global) y el tiempo de muestreo (recurso local). Para administrar el tiempo de procesamiento se hace uso del concepto de plataforma virtual v_i, la cual es una representación que caracteriza a un procesador virtual por su proporción de uso, es decir, por la fracción usada de la potencia disponible en dicho procesador. La proporción de uso se compone de dos parámetros: el periodo T y el presupuesto Q, de tal manera que dos procesadores virtuales con la misma proporción de uso pueden asignar tiempo de manera diferente. Por ejemplo, dos procesadores virtuales pueden tener una proporción de uso igual a 0,25, pero el primer procesador virtual tener un periodo de $4\,s$ y un presupuesto de $1\,s$, mientras que un segundo procesador virtual tener un periodo de $4\,ms$ y un presupuesto de $1\,ms$. En este caso, a pesar de tener la misma proporción de uso, el segundo procesador virtual es más adaptable en el sentido de que una aplicación puede ser atendida con mayor frecuencia y, por ello, progresar más uniformemente (Bini et al., 2011). Por otro lado, para asignar el tiempo de muestreo, se hace uso del nivel de servicio, donde este se toma como inversamente proporcional al tiempo de muestreo. Cada aplicación regula su nivel de servicio por medio de la función de emparejamiento (ver ecuación (4) más adelante). El *RM* que se presenta en este trabajo combina el modelo síncrono de Chasparis et al. (2016) con una propuesta para *CBS*, que serán descritos con mayor detalle en los siguientes apartados.

2.1. Modelo Chasparis síncrono

En este trabajo se considera un modelo síncrono, es decir, se considera que cada *RTA* sincroniza su nivel de servicio en la misma iteración en la que el *RM* asigna tiempo de procesamiento a la aplicación. En Chasparis et al. (2013) y Chasparis et al. (2016) se propone el modelo distribuido descrito por las ecuaciones (1)-(4).

$$\bar{v}_{i,k+1} = \Pi_{\bar{v}i}\left[\bar{v}_{i,k} + \epsilon F_{i,k}\right] \tag{1}$$

$$s_{i,k+1} = \Pi_{\mathcal{S}_i}[s_{i,k} + \epsilon f_{i,k}] \tag{2}$$

$$F_{i,k} \doteq -(U_s - \bar{v}_{i,k}))\lambda_i[f_{i,k}]\_ + \bar{v}_{i,k}\sum_{l\neq i}\lambda_l[f_{l,k}]\_ \tag{3}$$

$$f_{i,k} = \beta_i \frac{\kappa\bar{v}_{i,k}}{s_{i,k}} - 1 \tag{4}$$

donde

k : número de iteración; $k \in \mathbb{N}$.

i : número de aplicación; $i \in \mathbb{N}$..

$\bar{v}_{i,k}$: tamaño de la plataforma virtual normalizada asignada a la aplicación i en la iteración k, $\bar{v}_{i,k} = v_{i,k}/\kappa$; $\bar{v}_{i,k} \in \mathbb{R} : (0, 1]$, donde κ es el número de procesadores.

$F_{i,k}$: función de observación o equidad de la aplicación i que mide el efecto de las otras aplicaciones sobre ella en la iteración k; $F_{i,k} \in \mathbb{R} : [-1, 1]$.

$\bar{v}_s$: plataforma por distribuir entre las aplicaciones *soft*, tomando en cuenta que las aplicaciones están normalizadas $\bar{v}_s = 1 - \bar{v}_H$, donde $\bar{v}_H$ es la reservación para las aplicaciones *hard*.

ϵ : constante positiva, representa una pequeña cantidad de recursos que se asigna o se quita en cada iteración; $\epsilon \in \mathbb{R} : 0 < i < 1$.

$s_{i,k}$: nivel de servicio de la aplicación i en la iteración k; es un estado interno de la aplicación i, es dependiente de la plataforma virtual asignada a la aplicación i y sólo puede ser leído/escrito por la misma aplicación, $s_i \in \mathbb{R} : \underline{s_i} < s_i < \overline{s_i}$.

$f_{i,k}$: función de emparejamiento de la aplicación i en la iteración k, la cual indica si la aplicación está recibiendo suficientes o insuficientes recursos, el operador $[\ ]\_$ limita el rango de $[f_{i,k}]$ a $[-1, 0] \in \mathbb{R}$ y se define de la siguiente manera:

$$[x]_{\_} = \begin{cases} x & \text{si } x \leq 0 \\ 0 & \text{si } x > 0 \end{cases} \tag{5}$$

β_i : constante positiva dependiente de la aplicación i.

λ_i : prioridad de la aplicación i; $\lambda_i \in \mathbb{R} : [0, 1]$.

$\Pi_{\bar{V}_i}$ regresa a $\bar{v}_{i,k}$ al dominio de $\bar{V}_i$.

$\Pi_{\mathcal{S}_i}$ regresa a $\bar{s}_{i,k}$ al dominio de $\mathcal{S}_i$.

Este modelo no toma en cuenta la dinámica del *CBS*, cuya función consiste en asignar tiempo disponible a las tareas para su ejecución, y en caso de no existir tiempo disponible, de aplazar su deadline un periodo de servidor. Esta dinámica es de utilidad para la aplicación de diversas técnicas de control y se describe con mayor detalle en la siguiente sección.

3. Servidor de ancho de banda constante (*Constant Bandwidth Server* - CBS)

El *CBS* (Abeni and Buttazzo, 1998) es un mecanismo de servicio, con el cual se implementa una estrategia para reservar recursos de proceso. El *CBS* es caracterizado por los parámetros Q_s (presupuesto) y T_s (periodo), que garantiza que si $U_s = Q_s/T_s$ es la fracción del tiempo de procesador asignado a un servidor, su contribución a la utilización total no es mayor que U_s. El *CBS* está regido por las siguientes reglas.

1. Cuando un nuevo *job* entra al sistema, se le asigna una planificación de deadline adecuada para mantener la demanda dentro del ancho de banda reservado, y se inserta en la cola "*ready*" del *Earliest Deadline First* (EDF) (Horn, 1974).
2. Si un *job* trata de ejecutarse más de lo esperado, su *deadline* se pospone (lo que se consigue, por ejemplo, al decrecer su prioridad) para reducir la interferencia con las demás tareas. Al posponerse, la tarea sigue elegible para su ejecución.
3. Si un subconjunto de tareas es manejada por un solo servidor, todas las tareas en ese subconjunto compartirán el mismo ancho de banda, no hay aislamiento entre ellas. Todas las otras tareas en el sistema están protegidas contra sobrecostes que ocurran en el subsistema.

Las ecuaciones propuestas en este trabajo para describir la actualización del *deadline* y el presupuesto por cada *CBS* son:

$$c_{si,k+1} = c_{si,k} \rfloor (d_{si,k} - r_{i,k})\kappa\bar{v}_{i,k}c_{si,k}\lceil + Q_{si} \rfloor c_{si,k} - (d_{si,k} - r_{i,k})\kappa\bar{v}_{i,k}\lceil + Q_{si} \rfloor - c_{si,k}\lceil \tag{6}$$

$$d_{si,k+1} = d_{si,k} + T_{si} \rfloor (d_{si,k} - r_{i,k})\kappa\bar{v}_{i,k}c_{si,k}\lceil + T_{si} \rfloor - c_{si,k}\lceil \tag{7}$$

donde

si : identificador del servidor que atiende a la aplicación i; $si \in \mathbb{N} > 0$

$c_{si,k}$: presupuesto actual del servidor que atiende a la aplicación i; $c_{si,k} \in \mathbb{R} : 0 < c_{si,k} < Q_{si}$.

$d_{si,k}$: deadline del servidor que atiende a la aplicación i; $d_{s,k} \in \mathbb{R} > 0$.

T_{si} : periodo del servidor que atiende a la aplicación i; $T_{si} \in \mathbb{R} > 0$.

Q_{si} : máximo presupuesto del servidor que atiende a la aplicación i; $Q_{si} \in \mathbb{R} > 0$.

$\bar{v}_{i,k}$: proporción de uso asignada al servidor que atiende a la aplicación i y se define como $\bar{v}_{i,k} = Q_{si}/T_{si}$; $v_{i,k} \in \mathbb{R} > 0$.

$r_{i,k}$: tiempo de arribo de la aplicación i.

El operador binario $\rfloor x \lceil$ se define como

$$\rfloor x \lceil = \begin{cases} 0 & \text{si } x < 0 \\ 1 & \text{si } x \geq 0 \end{cases}$$

4. Integración del modelo Chasparis y el servidor de ancho de banda constante

En esta sección se describe el modelo propuesto en este trabajo que combina el modelo de Chasparis con el *CBS* propuesto en las ecuaciones(6-7).

Si se asume que el total de tiempo de procesamiento a distribuir entre las aplicaciones tipo *soft* es $\bar{v}_s$, este se debe distribuir entre las distintas aplicaciones en fracciones Q_{si} con un periodo T_{si} tal que:

$$\bar{v}_s = \sum_{i}^{n} \frac{Q_{si}}{T_{si}} \leq 1 - \bar{v}_H \tag{8}$$

donde $\bar{v}_H$ es la reservación para aplicaciones tipo *hard*. A cada plataforma virtual le corresponde una fracción del tiempo de procesamiento $\bar{v}_s$. En caso de que el periodo sea constante la relación entre el presupuesto Q_{si} y la plataforma virtual $\bar{v}_{i,k}$ es lineal.

Las relaciones entre ambas dinámicas se describen a continuación.

1. Se reciben las aplicaciones con los siguientes valores iniciales, $k = 0$, $\bar{v}_{i,0} = 0$, $s_{i,0} = \bar{s}_i$, $c_{si,0} = 0$, siendo $\bar{s}_i$ un máximo de nivel de servicio.
2. Debido a los valores iniciales, todas las aplicaciones empiezan con una función de emparejamiento $f_{i,0} = -1$ y por consecuencia un nivel de equidad nominal positivo $F_{i,0} > 0$, dependiente de las prioridades λ_i.
3. Se calcula la actualización de la plataforma virtual $\bar{v}_{i,k+1}$.

4. A partir de la plataforma virtual $\bar{v}_{i,k}$ y del tiempo de ejecución nominal C_i, se calcula el periodo y el presupuesto máximo del *CBS*; $T_{si} = C_i/\bar{v}_{i,k}$ y $Q_{si} = \kappa\, \bar{v}_{i,k} T_{si}$.
5. Se actualiza el deadline absoluto $d_{si,k+1} = d_{si,k} + T_{si} \rfloor (d_{si,k} - r_{i,k})\kappa\bar{v}_{i,k} c_{si,k} \lceil + T_{si} \rfloor - c_{si,k} \lceil$, siendo $d_{si,0} = r_i$.
6. Se actualiza el presupuesto de la aplicación $c_{si,k+1} = c_{si,k} \rfloor (d_{si,k} - r_{i,k})\kappa\bar{v}_{i,k} c_{si,k} \lceil + Q_{si} \rfloor c_{si,k} - (d_{si,k} - r_{i,k})\kappa\bar{v}_{i,k} \lceil + Q_{si} \rfloor - c_{si,k} \lceil$.
7. En caso de que sí se lleve a cabo la ejecución de la aplicación, el presupuesto decrece a $c_{si,k} - t$, donde t, es el tiempo de ejecución de la aplicación.
8. Se actualiza el índice de tiempo $k \leftarrow k + 1$ y se repite el ciclo a partir de 3.

En consecuencia, el modelo de Chasparis extendido propuesto en este trabajo tendrá la siguiente forma:

$$\bar{v}_{i,k+1} = \bar{v}_{i,k} + \epsilon F_{i,k} \tag{9}$$

$$s_{i,k+1} = \bar{s}_{i,k} + \epsilon f_{i,k} \tag{10}$$

$$c_{si,k+1} = c_{si,k} \rfloor (d_{si,k} - r_{i,k})\kappa\bar{v}_{i,k} c_{si,k} \lceil + Q_{si} \rfloor c_{si,k} - (d_{si,k} - r_{i,k})\kappa\bar{v}_{i,k} \lceil + Q_{si} \rfloor - c_{si,k} \lceil \tag{11}$$

$$d_{si,k+1} = d_{si,k} + T_{si} \rfloor (d_{si,k} - r_{i,k})\kappa\bar{v}_{i,k} c_{si,k} \lceil + T_{si} \rfloor - c_{si,k} \lceil \tag{12}$$

Para realizar este análisis se supone que

$$\rfloor (d_{si,k} - r_{i,k})\kappa\bar{v}_{i,k} c_{si,k} \lceil \simeq 1 \tag{13}$$

restando el estado anterior al estado actual se obtiene:

$$\bar{v}_{i,k+1} - \bar{v}_{i,k} = \epsilon F_{i,k} \tag{14}$$

$$s_{i,k+1} - \bar{s}_{i,k} = \epsilon f_{i,k} \tag{15}$$

$$c_{si,k+1} - c_{si,k} = Q_{si} \rfloor c_{si,k} - (d_{si,k} - r_{i,k})\kappa\bar{v}_{i,k} \lceil + Q_{si} \rfloor - c_{si,k} \lceil \tag{16}$$

$$d_{si,k+1} - d_{si,k} = T_{si} \rfloor (d_{si,k} - r_{i,k})\kappa\bar{v}_{i,k} c_{si,k} \lceil + T_{si} \rfloor - c_{si,k} \lceil \tag{17}$$

Dividiendo por un diferencial $\Delta\tau$.

$$\frac{\bar{v}_{i,k+1} - \bar{v}_{i,k}}{\Delta\tau} = \frac{\epsilon F_{i,k}}{\Delta\tau} \tag{18}$$

$$\frac{s_{i,k+1} - \bar{s}_{i,k}}{\Delta\tau} = \frac{\epsilon f_{i,k}}{\Delta\tau} \tag{19}$$

$$\frac{c_{si,k+1} - c_{si,k}}{\Delta\tau} = \frac{Q_{si} \rfloor c_{si,k} - (d_{si,k} - r_{i,k})\kappa\bar{v}_{i,k} \lceil + Q_{si} \rfloor - c_{si,k} \lceil}{\Delta\tau} \tag{20}$$

$$\frac{d_{si,k+1} - d_{si,k}}{\Delta\tau} = \frac{T_{si} \rfloor (d_{si,k} - r_{i,k})\kappa\bar{v}_{i,k} c_{si,k} \lceil + T_{si} \rfloor - c_{si,k} \lceil}{\Delta\tau} \tag{21}$$

Utilizando la definición de derivada se obtiene el siguiente sistema de ecuaciones diferenciales dependientes de una variable temporal τ para realizar un análisis de estabilidad.
Como ϵ es una constante distinta de cero que está multiplicando a la parte dinámica de la ecuación, se descarta del análisis, debido a que no afecta la ubicación de los puntos de equilibrio, el sistema de ecuaciones diferenciales es el siguiente:

$$\dot{\bar{v}}_i(\tau) = -(\bar{v}_s - \bar{v}_i(\tau))\lambda_i [f_i(\tau)]_- + \bar{v}_i(\tau) \sum_{j \neq i} \lambda_j [f_j(\tau)]_- \tag{22}$$

$$\dot{s}_i(\tau) = \beta_i \frac{\kappa\bar{v}_i(\tau)}{s_i(\tau)} - 1 \tag{23}$$

$$\dot{c_{si}}(\tau) = Q_{si} \rfloor c_{si}(\tau) - (d_{si}(\tau) - r_i(\tau))\kappa\bar{v}_i(\tau) \lceil + Q_{si} \rfloor - c_{si}(\tau) \lceil \tag{24}$$

$$\dot{d}_{si}(\tau) = T_{si} \rfloor (d_{si}(\tau) - r_i(\tau))\kappa\bar{v}_i c_{si}(\tau) \lceil + T_{si} \rfloor - c_{si}(\tau) \lceil \tag{25}$$

donde todas las ecuaciones dependen de τ. Se definen las actualizaciones de los estados

$$\Phi_i(\kappa\bar{\mathbf{v}}(\tau), \mathbf{s}(\tau)) = -(\bar{v}_s - \bar{v}_i(\tau))\lambda_i [f_i(\tau)]_- + \bar{v}_i(\tau) \sum_{j \neq i} \lambda_j [f_j(\tau)]_- \tag{26}$$

$$\varphi_i(\kappa\bar{\mathbf{v}}(\tau), \mathbf{s}(\tau)) = \beta_i \frac{\kappa\bar{v}_i(\tau)}{s_i(\tau)} - 1 \tag{27}$$

$$\Lambda_i(d_{si}(\tau), c_{si}(\tau), \bar{v}_i(\tau)) = Q_{si} \rfloor c_{si}(\tau) - (d_{si}(\tau) - r_i(\tau))\kappa\bar{v}_i(\tau) \lceil + Q_{si} \rfloor - c_{si}(\tau) \lceil \tag{28}$$

$$\Gamma_i(c_{si}(\tau), \bar{v}_i(\tau)) = T_{si} \rfloor (d_{si}(\tau) - r_i(\tau))\kappa\bar{v}_i c_{si}(\tau) \lceil + T_{si} \rfloor - c_{si}(\tau) \lceil \tag{29}$$

El *RM* se reescribe como un sistema de ecuaciones diferenciales ordinarias no lineales, cuyo punto de equilibrio se encuentra cuando Φ_i y φ_i , Γ_i y Λ_i son iguales a cero en el sistema (30)

$$\begin{bmatrix} \dot{\bar{v}}_i(\tau) \\ \dot{s}_i(\tau) \\ \dot{c_{si}}(\tau) \\ \dot{d}_{si}(\tau) \end{bmatrix} = \begin{bmatrix} \Phi_i(\kappa\bar{\mathbf{v}}(\tau), \mathbf{s}(\tau)) \\ \varphi_i(\kappa\bar{v}_i(\tau), s_i(\tau)) \\ \Lambda_i(d_{si}(\tau), c_{si}(\tau), \bar{v}_i(\tau)) \\ \Gamma_i(c_{si}(\tau), \bar{v}_i(\tau)) \end{bmatrix} + Z \tag{30}$$

donde $Z = [0\ 0\ -d^*_{si}\ 0]^T$ es un vector de corrección que traslada las soluciones al dominio de las soluciones factibles.
El punto de equilibrio se encuentra cuando las variables tienen los siguientes valores: $d_{si} = d^*_{si}$, $c_{si} = Q_{si}$, $\mathbf{v} = \mathbf{v}^*$ y $s_i = \underline{s_i}$. Este punto de equilibrio indica que el *deadline* d^*_{si} es el máximo admisible, el presupuesto está lleno en espera de nuevas tareas, la asignación de la plataforma virtual asignada $\mathbf{v}^*$ es equitativa (función de emparejamiento $f_i = 0$, función de equidad nominal $F_i = 0$), es decir, las tareas tienen un grado de uso en el que cumplen con sus *deadlines*, lo que se traduce en menor menor espera y menor retardo en el *NCS*. Por último el nivel de servicio llega a su nivel inferior $\underline{s_i}$, lo cual indica que la tarea es capaz de sacrificar nivel de servicio, sin disminuir su desempeño.

5. Linealización

Se linealiza el sistema alrededor de su punto de equilibrio utilizando el Jacobiano evaluado en dicho punto.

$$J = \begin{bmatrix} \frac{\partial\Phi_i}{\partial d_{si}} & \frac{\partial\Phi_i}{\partial c_{si}} & \frac{\partial\Phi_i}{\partial\bar{v}_i} & \frac{\partial\Phi_i}{\partial s_i} \\ \frac{\partial\varphi_i}{\partial d_{si}} & \frac{\partial\varphi_i}{\partial c_{si}} & \frac{\partial\varphi_i}{\partial\bar{v}_i} & \frac{\partial\varphi_i}{\partial s_i} \\ \frac{\partial\Gamma_i}{\partial d_{si}} & \frac{\partial\Gamma_i}{\partial c_{si}} & \frac{\partial\Gamma_i}{\partial\bar{v}_i} & \frac{\partial\Gamma_i}{\partial s_i} \\ \frac{\partial\Lambda_i}{\partial d_{si}} & \frac{\partial\Lambda_i}{\partial c_{si}} & \frac{\partial\Lambda_i}{\partial\bar{v}_i} & \frac{\partial\Lambda_i}{\partial s_i} \end{bmatrix} \tag{31}$$

$$= \begin{bmatrix} 0 & 0 & 1 - \frac{\beta\kappa}{s_i} & -(v^*_i - U_s) \\ 0 & 0 & \frac{\beta_i\kappa}{s_i} & -\frac{1}{s_i} \\ -1 & \frac{1}{\bar{v}^*_i} & \frac{-q_i}{\bar{v}^{*2}_i} & 0 \\ 0 & Q_{si} & 0 & 0 \end{bmatrix} \tag{32}$$

La linealización tiene un punto de equilibrio es estable solamente cuando a d_{si} se le agrega el término de corrección a través del vector Z mencionado arriba, que garantiza la convergencia de las trayectorias a una distribución equitativa.

6. Modelación difusa

Un *RM* como el introducido en este trabajo incorpora términos no lineales, cuya presencia dificulta el diseño de esquemas de control. Una posibilidad para tratar con estas no linealidades es aprovechar la capacidad que tienen las redes neuronales artificiales y los sistemas difusos para modelar sistemas no lineales, pues pueden aproximar con precisión arbitrariamente pequeña los términos no lineales, si cuentan con suficientes neuronas ocultas y datos de entrenamiento o reglas difusas, respectivamente (Boutalis et al., 2014).

Aquí se plantea utilizar un modelo difuso tipo Takagi-Sugeno (Tanaka et al., 1998) el cual está descrito por una serie de reglas **IF − THEN** (ver ejemplos en las Tablas 1 y 2 más adelante). Este *RM* difuso será capaz de aproximar las no linealidades del modelo Chasparis extendido mediante un conjunto de sistemas lineales. Como en todo sistema difuso, el estado actual del modelo tiene asignado un nivel de membresía a cada elemento del conjunto de sistemas no lineales. En todos los casos, las funciones de membresía utilizadas serán de forma triangular. Así la combinación modelo difuso-controlador puede, a partir de cualquier distribución inicial arbitraria de recursos, llevar esta hacia una distribución justa, sin que se pierda la estabilidad en lazo cerrado del sistema aproximado.

Además de la aproximación difusa, se propone distribuir el modelo de Chasparis extendido en tres subsistemas (Figura 2): i) ajuste de servicio; ii) distribución de plataforma virtual; iii) actualización del *deadline* y presupuesto. Estos subsistemas se pueden tratar como tres sistemas independientes para diseñar los esquemas de control que garanticen su estabilidad en lazo cerrado.

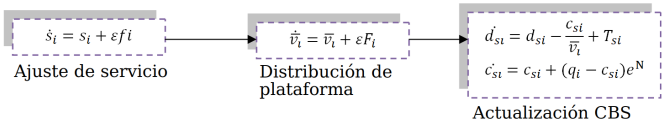

Figura 2: El sistema se divide en tres subsistemas, los cuales están enlazados por un mínimo de información.

6.1. Modelación difusa del nivel de plataforma virtual

Para la distribución de la plataforma virtual se necesita conocer el valor de la función de emparejamiento f_i de cada aplicación, para así poder calcular la función equidad nominal F_i; si se desarrolla dicha función se obtiene:

$$\dot{\bar{v}}_i = \bar{v}_i + \epsilon F_i = \bar{v}_i + \epsilon \left\{ -(U_s - \bar{v}_i)\lambda_i [f_i]_- + \bar{v}_i \sum_{j \neq i} \lambda_j [f_j]_- \right\} \tag{33}$$

Se define la variable premisa

$$z_{v_i} = \epsilon \left(-(U_s - \bar{v}_i)[f_i]_- + \frac{\bar{v}_i}{\lambda_i} \sum_{j}^{n} \lambda_j [f_j]_- \right) \tag{34}$$

y las funciones de membresía

$$M_{vi_1} = 1 - \frac{z_{v_i} - z_{vi_{max}}}{z_{vi_{min}} - z_{vi_{max}}} \tag{35}$$

$$M_{vi_2} = \frac{z_{v_i} - z_{vi_{max}}}{z_{vi_{min}} - z_{vi_{max}}} \tag{36}$$

Las cuales están relacionadas con las declaraciones:

- M_{vi_1} Recursos insuficientes para aplicación i
- M_{vi_2} Recursos suficientes para aplicación i

y con los valores máximo vi_{max} y mínimo vi_{min} de la plataforma virtual $\bar{v}_i$.

6.2. Modelación difusa del nivel de servicio

El ajuste del nivel de servicio también se lleva a cabo dentro de cada aplicación pues, como se mencionó, este depende de la naturaleza de la misma. Para ello, se deben definir niveles de servicio máximo si_{max} y mínimo si_{min}, de manera que el nivel de servicio esté siempre entre ellos. El nivel de servicio se calcula de la siguiente ecuación

$$\dot{s}_i = s_i + \epsilon f_i = s_i + \left[\epsilon \left(\frac{\beta_i K \bar{v}_i}{s_i} - 1 \right) \right]_- \tag{37}$$

donde se puede notar que para su cálculo la aplicación i solamente requiere información local. Se define como variable premisa para cada aplicación:

$$z_{s_i} = \epsilon \left(\frac{\beta_i K}{s_i} - 1 \right) \tag{38}$$

De esta variable premisa, se definen dos funciones de membresía que representan los extremos de las situaciones en las que se puede encontrar el nivel de servicio y que están descritas por las siguientes ecuaciones

$$M_{si_1} = 1 - \frac{z_{s_i} - z_{si_{max}}}{z_{si_{min}} - z_{si_{max}}} \tag{39}$$

$$M_{si_2} = \frac{z_{s_i} - z_{si_{max}}}{z_{si_{min}} - z_{si_{max}}} \tag{40}$$

Cada una de estas funciones de membresía está relacionadas con las declaraciones:

- M_{si_1} Máximo servicio en la aplicación i
- M_{si_2} Mínimo servicio en la aplicación i.

La relación entre las reglas difusas y los sistemas lineales asociados a cada una se ilustra en la Tabla 1, donde los sistemas $A_{vj}x_v + B_v u_v$ tienen la estructura de la ecuación (41), con i el número de la aplicación y n el número total de estas. Como el cálculo de la plataforma virtual $\bar{v}_i$ y del nivel de servicio se realiza de manera distribuida, el número de reglas y sistemas lineales asociados es $4n$. En el ejemplo de la Tabla 1, $n = 3$, por lo tanto, si se combinan las reglas del nivel de plataforma virtual y las del nivel de servicio, se obtienen 12 reglas asociadas a 12 sistemas lineales.

Tabla 1: Relaciones entre las funciones de membresía $M_{vi_1}(z_{v_i})$, $M_{vi_2}(z_{v_i})$, $M_{si_1}(z_{s_i})$, $M_{si_2}(z_{s_i})$; $i = 1, 2, 3$, y w_{vj}, $j = 1, 2, 3..,12$ es la cuantificación de cada combinación. La forma de los sistemas lineales $A_{vj}x_v + B_v u_v$ es presentada en la ecuación (41)

$w_{vj}(z)$	IF	THEN $A_{vj}x_v + B_v u_v$
$w_{v1}(z)$	$M_{v1_1}(z_{v_1})$	$A_{v1}x_v + B_v u_v$
$w_{v2}(z)$	$M_{v1_2}(z_{v_1})$	$A_{v2}x_v + B_v u_v$
$w_{v3}(z)$	$M_{v2_1}(z_{v_2})$	$A_{v3}x_v + B_v u_v$
$\vdots$	$\vdots$	$\vdots$
$w_{v11}(z)$	$M_{s3_1}(z_{s_3})$	$A_{v11}x_v + B_v u_v$
$w_{v12}(z)$	$M_{s3_2}(z_{s_3})$	$A_{v12}x_v + B_v u_v$

$$\begin{bmatrix} \dot{\bar{v}}_1 \\ \vdots \\ \dot{\bar{v}}_n \\ \dot{s}_1 \\ \vdots \\ \dot{s}_n \\ \dot{\lambda}_1 \\ \vdots \\ \dot{\lambda}_n \end{bmatrix} = \begin{bmatrix} z_{v_1} & \cdots & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & z_{v_n} & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & z_{s_1} & \cdots & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \ddots & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & z_{s_n} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \bar{v}_1 \\ \vdots \\ \bar{v}_n \\ s_i \\ \vdots \\ s_n \\ \lambda_i \\ \vdots \\ \lambda_n \end{bmatrix} + \begin{bmatrix} 0 & \cdots & 0 \\ 0 & \cdots & 0 \\ 0 & \cdots & 0 \\ & \vdots & \\ 1 & \cdots & 0 \\ 0 & \ddots & 0 \\ 0 & \cdots & 1 \end{bmatrix} \begin{bmatrix} u_{v1} \\ \vdots \\ u_{vn} \end{bmatrix} \tag{41}$$

Para garantizar la estabilidad de lazo cerrado se deben proponer las F_i y encontrar una matriz $P > 0$ que cumpla la siguiente desigualdad para todas las matrices A_i.

$$G_{ii} = A_i - B_i F_i \tag{42}$$

$$0 > G_{ii}^T P G_{ii} - P \tag{43}$$

6.3. Modelación difusa del nivel de CBS

El *CBS* actualiza dos estados, el *deadline* d_{si} y el presupuesto c_{si}, de acuerdo con las siguientes ecuaciones.

$$\dot{c}_{si} = Q_{si} \lfloor c_{si} - (d_{si} - r_i)\kappa\bar{v}_i \rceil + Q_{si} \lfloor -c_{si} \rceil \tag{44}$$

$$\dot{d}_{si} = T_{si} \lfloor (d_{si} - r_i)\kappa\bar{v}_i c_{si} \rceil + T_{si} \lfloor -c_{si} \rceil \tag{45}$$

Al ser dos ecuaciones se definen dos variables premisa por aplicación, haciendo que las combinaciones sean 2^{n*2}, donde n es el número de aplicaciones.

$$z_{cs_i} = \kappa T_{si} \lfloor c_{si} - (d_{si} - r_i)\kappa\bar{v}_i \rceil \tag{46}$$

$$z_{ds_i} = \frac{1}{\kappa\bar{v}_i} \lfloor (d_{si} - r_i)\kappa\bar{v}_i c_{si} \rceil \tag{47}$$

Las funciones de membresía están definidas por las ecuaciones (48-51)

$$M_{csi_1} = 1 - \frac{z_{cs_i} - z_{csi_{max}}}{z_{csi_{min}} - z_{csi_{max}}} \tag{48}$$

$$M_{csi_2} = \frac{z_{cs_i} - z_{csi_{max}}}{z_{csi_{min}} - z_{csi_{max}}} \tag{49}$$

$$M_{dsi_1} = 1 - \frac{z_{ds_i} - z_{dsi_{max}}}{z_{dsi_{min}} - z_{dsi_{max}}} \tag{50}$$

$$M_{dsi_2} = \frac{z_{ds_i} - z_{dsi_{max}}}{z_{dsi_{min}} - z_{dsi_{max}}} \tag{51}$$

donde csi_{max}, csi_{min}, dsi_{max} y dsi_{min} son los límites máximo y mínimo de cs_i y ds_i, respectivamente. Cada una ligada a las siguientes declaraciones:

- M_{csi_1} presupuesto insuficientes para aplicación i.
- M_{csi_2} Rellenar presupuesto para aplicación i.
- M_{dsi_1} No se requiere actualizar el *deadline* de la aplicación i.
- M_{dsi_2} Actualizar el *deadline* de la aplicación i..

Un ejemplo de la relación entre las reglas difusas y los sistemas lineales se muestra en la Tabla 2 para un sistema con $n = 3$ aplicaciones.

Los sistemas lineales $A_{ci}x_c + B_c u_c$ tienen la forma de la ecuación (52):

$$\begin{bmatrix} \dot{c}_{s_i} \\ \vdots \\ \dot{c}_{s_n} \\ \dot{d}_{s_i} \\ \vdots \\ \dot{d}_{s_n} \end{bmatrix} = \begin{bmatrix} 1 & \ldots & 0 & z_{dc_i} & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 1 & 0 & \cdots & z_{dc_n} \\ 0 & 0 & 0 & z_{d_i} & \cdots & 0 \\ 0 & 0 & 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & 0 & \cdots & z_{d_n} \end{bmatrix} \begin{bmatrix} c_{s_i} \\ \vdots \\ c_{s_n} \\ d_{s_i} \\ \vdots \\ d_{s_n} \end{bmatrix} + \begin{bmatrix} 0 & \cdots & 0 \\ 0 & \cdots & 0 \\ 0 & \cdots & 0 \\ 1 & \cdots & 0 \\ 0 & \ddots & 0 \\ 0 & \cdots & 1 \end{bmatrix} \begin{bmatrix} u_i \\ \vdots \\ u_n \end{bmatrix} \tag{52}$$

De la misma manera que en los dos subsistemas anteriores, se requiere cumplir con la condición (42) para garantizar estabilidad en lazo cerrado.

Tabla 2: Combinaciones entre las funciones de membresía $M_{csi_1}(z_{cs_i})$, $M_{csi_2}(z_{cs_i})$, $M_{dsi_1}(z_{ds_i})$, $M_{dsi_2}(z_{ds_i})$; $i = 1, 2, 3$, $(\cdot)$ representa la función lógica *AND* y w_{cj}, $j = 1, 2, 3..,64$ es la cuantificación de cada combinación. La forma de los sistemas lineales $A_{cj}x_c + B_cu_c$ es presentada en la ecuación (52)

$w_{csj}(z)$	IF	THEN $A_{cj}x_c + B_cu_c$
$w_{cs1}(z)$	$M_{cs1_1}(z_{cs_1}) \cdot M_{cs2_1}(z_{cs_2}) \cdot M_{cs3_1}(z_{cs_3}) \cdot M_{ds1_1}(z_{ds_1}) \cdot M_{ds2_1}(z_{ds_2}) \cdot M_{ds3_1}(z_{ds_3})$	$A_{c1}x_c + B_cu_c$
$w_{cs2}(z)$	$M_{cs1_1}(z_{cs_1}) \cdot M_{cs2_1}(z_{cs_2}) \cdot M_{cs3_1}(z_{cs_3}) \cdot M_{ds1_1}(z_{ds_1}) \cdot M_{ds2_1}(z_{ds_2}) \cdot M_{ds3_2}(z_{ds_3})$	$A_{c2}x_c + B_cu_c$
$w_{cs3}(z)$	$M_{cs1_1}(z_{cs_1}) \cdot M_{cs2_1}(z_{cs_2}) \cdot M_{cs3_1}(z_{cs_3}) \cdot M_{ds1_1}(z_{ds_1}) \cdot M_{ds2_1}(z_{ds_2}) \cdot M_{ds3_1}(z_{ds_3})$	$A_{c3}x_c + B_cu_c$
$\vdots$	$\vdots$	$\vdots$
$w_{cs63}(z)$	$M_{cs1_2}(z_{cs_1}) \cdot M_{cs2_2}(z_{cs_2}) \cdot M_{cs3_2}(z_{cs_3}) \cdot M_{ds1_2}(z_{ds_1}) \cdot M_{ds2_2}(z_{ds_2}) \cdot M_{ds3_1}(z_{ds_3})$	$A_{c63}x_c + B_cu_c$
$w_{cs64}(z)$	$M_{cs1_2}(z_{cs_1}) \cdot M_{cs2_2}(z_{cs_2}) \cdot M_{cs3_2}(z_{cs_3}) \cdot M_{ds1_2}(z_{ds_1}) \cdot M_{ds2_2}(z_{ds_2}) \cdot M_{ds3_2}(z_{ds_3})$	$A_{c64}x_c + B_cu_c$

7. Resultados de simulación

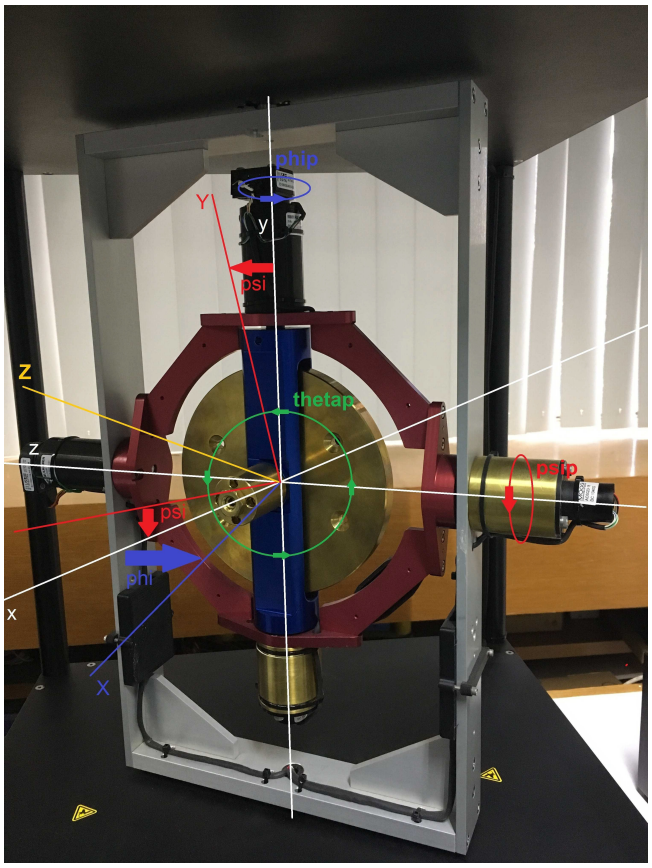

Figura 3: Giroscopio, marco de referencia: *theta* = θ, *psi* = ψ, *phi* = ϕ, *thetap* = $\dot{\theta}$, *psip* = $\dot{\psi}$, *phip* = $\dot{\phi}$.

Para probar el funcionamiento del *RM* y del *CBS* se recurre a la simulación de un sistema de control en red (*Networked control system - NCS*) de configuración directa (Mahmoud et al., 2013) aplicado al modelo un giroscopio de tres grados de libertad (ver Figura 3).

La meta del *NCS* es controlar dos de los tres grados de libertad del giroscopio, que corresponden con los ángulos ϕ y ψ en el marco de referencia mostrado en la Figura 3 y cuyo modelo dinámico está dado por (Robert H. Cannon, 2003)

$$M_b = J_y\ddot{\phi} - J_x^d\dot{\theta}\dot{\psi}\cos(\phi) + (J_z - J_x)\dot{\psi}^2\sin(\phi)\cos(\phi) \quad (53)$$

$$M_r = (J_z^r + J_z\cos^2(\phi) + J_x\sin^2(\phi))\ddot{\phi} + J_x^d\dot{\theta}\dot{\phi}\cos(\phi) + 2(J_x - J_z)\dot{\psi}\dot{\phi}\sin(\phi)\cos\phi \quad (54)$$

Los términos que componen las ecuaciones de movimiento (53-54) y el valor de sus parámetros dados por el fabricante (Quanser, 2012) son:

$\dot{\theta}$: velocidad angular del disco alrededor de su propio eje de rotación x. $\dot{\theta} = 150rad/s$.

ϕ : posición angular de la suspensión cardán azul sobre y.

ψ : posición angular de la suspensión cardán roja sobre Z.

J_z^r : momento de inercia de la suspensión cardán roja sobre el eje Z. $J_z^r = 0{,}0342kgm^2$

J_z^d : momento de inercia del disco sobre el eje x. $J_z^d = 0{,}0056kgm^2$

J_x : momento de inercia del disco y la suspensión cardán azul alrededor del eje x. $J_x = 0{,}0074kgm^2$

J_y : momento de inercia del disco y la suspensión cardán azul alrededor del eje y. $J_y = 0{,}0026kgm^2$

J_z : momento de inercia del disco y la suspensión cardán azul alrededor del eje z. $J_z = 0{,}0056kgm^2$

M_b : par externo total alrededor del eje de rotación de la suspensión cardán azul.

M_r : par externo total alrededor del eje de rotación de la suspensión cardán roja.

El control en lazo cerrado del giroscopio se realiza también a través de un sistema difuso, los detalles de su diseño se pueden encontrar en Aparicio (2017).

En el *NCS* se establecen tres aplicaciones tipo *soft*, a cada de las cuales se le asignan tareas distintas, según se describe en la Figura 4. La *APP*1 se encarga de las tareas de lectura de los sensores, la *APP*2 del cálculo de la señal de control y la *APP*3 de las tareas de escritura de los actuadores. El *RM* se implementa en cada una de las aplicaciones, mientras el *CBS* se implementó mediante una tarea centralizada tipo *hard*. Además de realizar las tareas que tienen asignadas, las aplicaciones deben enviar por la red el valor instantáneo de sus funciones de emparejamiento f_i ; $i = 1, 2, 3$.

La meta del sistema de control es conseguir que los ángulos ϕ y ψ sigan el par de señales sinusoidales mostradas en la Figura 5, donde $\phi = xq1$ y $\psi = xq2$.

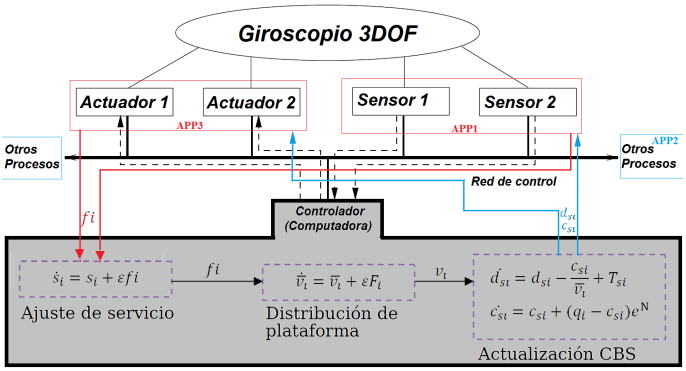

Figura 4: Sistema *NCS* configuración directa, conectado al administrador de recursos, se observa que la única información que se recibe de las aplicaciones es el valor de la función de emparejamiento f_i .

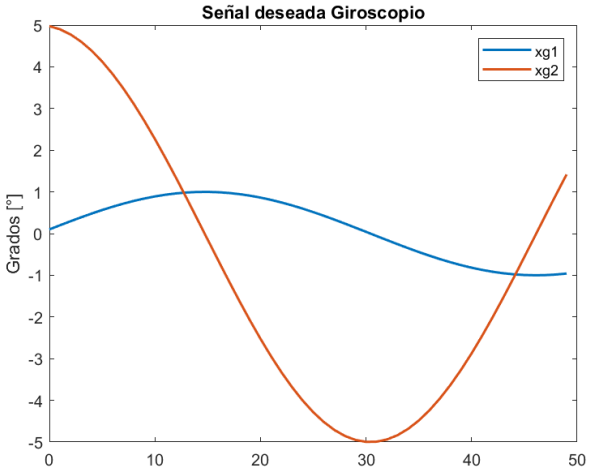

Figura 5: Señal deseada del giroscopio.

Para las simulaciones se usan las siguientes condiciones iniciales $\bar{v}_{1,0} = 0{,}1$, $\bar{v}_{2,0} = 0{,}5$, $\bar{v}_{3,0} = 0{,}4$, $s_{1,0} = 3$, $s_{2,0} = 1$, $s_{3,0} = 1$, $\epsilon = 0{,}1$, $\lambda_1 = 0{,}5$, $\lambda_2 = 0{,}5$, $\lambda_3 = 0{,}5$ y un periodo de servidor de $T_s = 0{,}3$, se agrega un retardo en la comunicación de $10^{-3}s$ que simula los retardos en la red.

Se presentan resultados de tres simulaciones distintas. En la primera se simula el uso de un solo procesador $\kappa = 1$, que debe ejecutar las tres aplicaciones *soft* y reservar tiempo para las aplicaciones *hard*, que utilizan 0,2 de la capacidad de procesamiento v_H. Por ello, la plataforma a compartir es $v_s = 0{,}8$.

La evolución de la plataforma virtual v_i en el tiempo se puede observar en la Figura 6, la que incluye también una línea punteada que indica el nivel mínimo de las plataformas virtuales que garantiza el apropiado desempeño de las tares que tienen asignadas.Puede notarse que las APP2 y APP3 tiene una plataforma virtual por debajo del valor mínimo necesario. La Figura 7 muestra la evolución del presupuesto de la aplicaciones, donde pueden observarse como este decrece y al llegar a cero vuelve al ser rellenado. En las Figuras 8 y 9 se muestran, respectivamente, la evolución de las funciones de emparejamiento y equidad nominal de cada aplicación. Aquí puede notarse la convergencia de todas las aplicaciones a los valores correctos.

Finalmente la Figura 10 muestra el error de seguimiento, el cual confirma la incapacidad del *NCS* para resolver el problema de control del giroscopio.

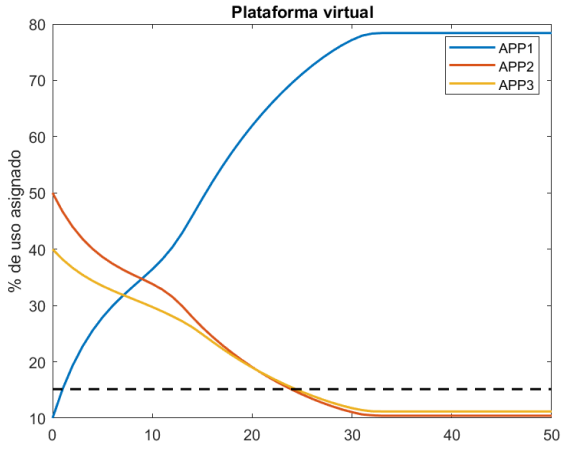

Figura 6: Plataforma virtual $v_{i,k}$,caso 1.

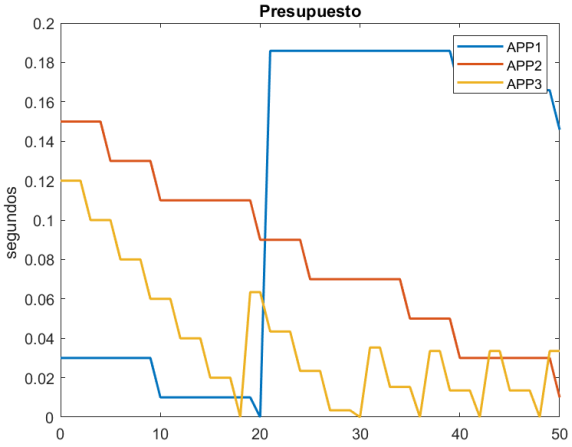

Figura 7: Evolución del presupuesto $c_{si,k}$, caso 1.

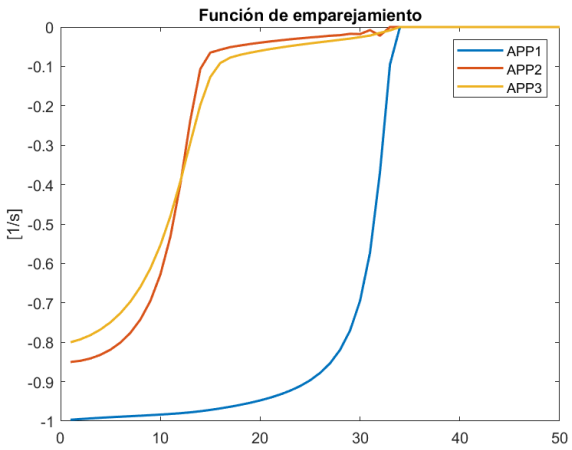

Figura 8: Función de emparejamiento $f_{i,k}$, caso 1.

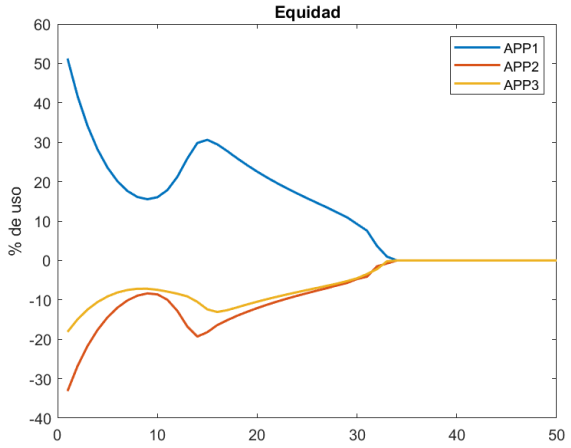

Figura 9: Función de equidad nominal $F_{i,k}$, caso 1.

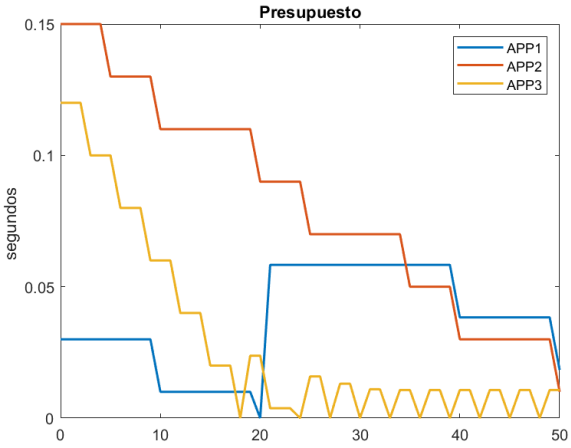

Figura 12: Evolución del presupuesto $c_{si,k}$, caso 2.

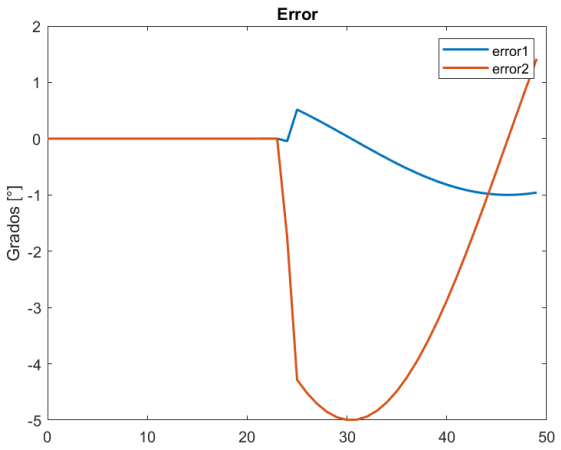

Figura 10: Señal de error del giroscopio, caso 1.

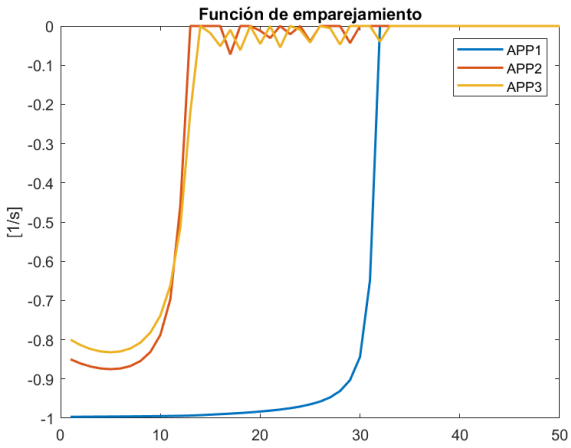

Figura 13: Función de emparejamiento $f_{i,k}$, caso 2.

Para la segunda simulación se sigue usando un procesador, pero los recursos a distribuir son menores pues se elige $v_s = 0{,}3$. En las Figuras 11-15 se puede observar que dado que el porcentaje disponible para uso es menor que en la primera simulación, los errores se incrementan.

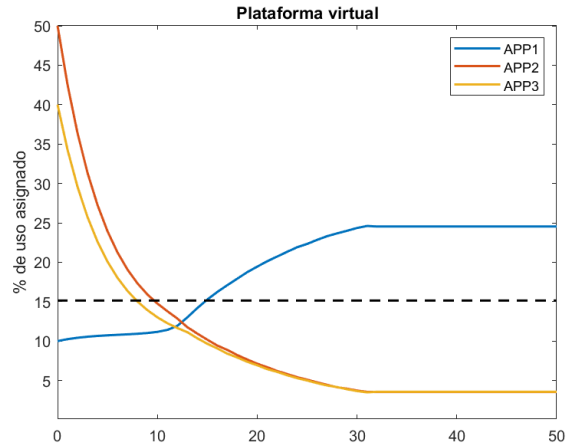

Figura 11: Plataforma virtual $v_{i,k}$, caso 2.

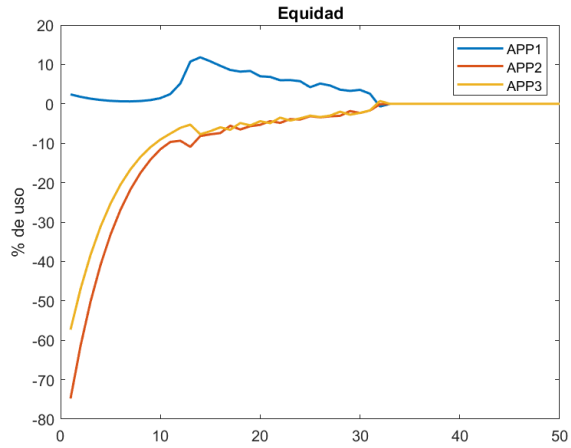

Figura 14: Función de equidad nominal $F_{i,k}$, caso 2.

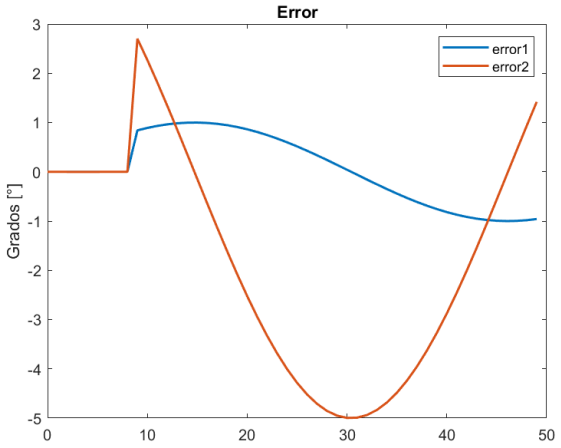

Figura 15: Señal de error del giroscopio, caso 2.

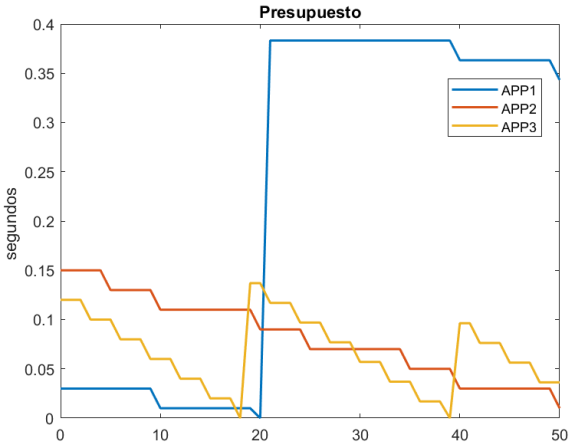

Figura 17: Evolución del presupuesto $c_{si,k}$, caso 3.

Para la tercera simulación, el número de procesadores simulados se aumenta a dos, $\kappa = 2$, con las mismas condiciones iniciales que en el primer caso. Al aumentar el número de procesadores, se aumenta la cantidad de recursos globales, por lo que ahora las aplicaciones se encuentran arriba del mínimo de plataforma virtual como se muestra en la Figura 16. El aumento en el número de procesadores también implica que existe mayor presupuesto para las aplicaciones, por lo que estas tardan más tiempo en consumirlo, como se observa en la Figura 17. La evolución de las funciones de emparejamiento y de equidad nominal muestra un comportamiento similar a los experimentos anteriores, como se muestra en las Figuras 18 y 19, respectivamente.

El desempeño del control del giroscopio se muestra en la Figura 20 que ilustra un mucho mejor seguimiento de las señales sinusoidales. Existe un error alrededor de los 24 s con una magnitud muy pequeña, de alrededor de 3×10^{-7} que es debido al cálculo de la señal de control y no al *RM*, pues, como se mostró en la Figura 16, este asignó los recursos necesarios para el buen funcionamiento de las aplicaciones.

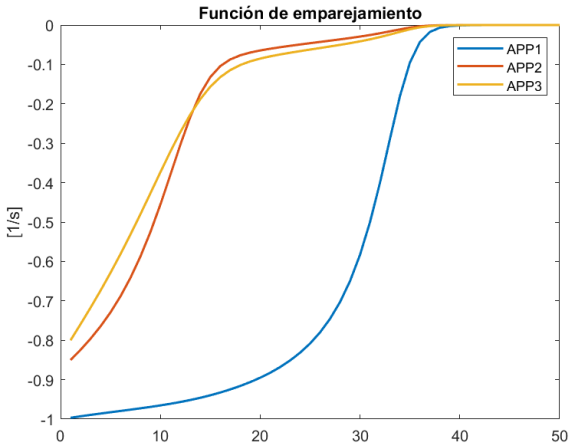

Figura 18: Función de emparejamiento $f_{i,k}$, caso 3.

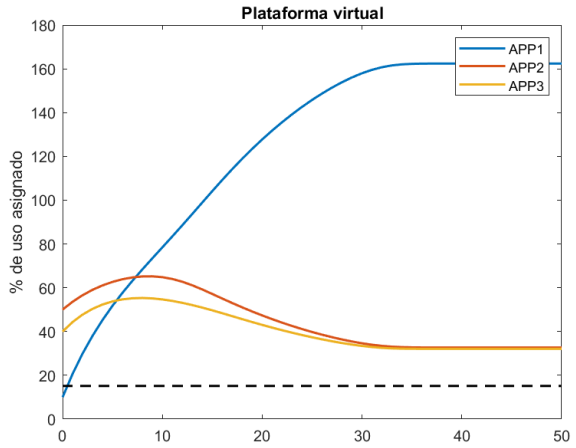

Figura 16: Plataforma virtual $v_{i,k}$,caso 3.

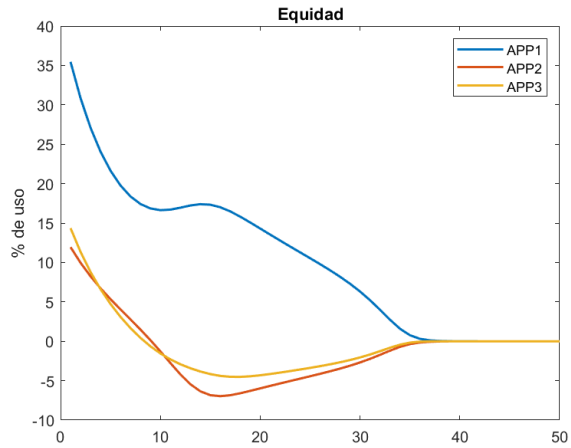

Figura 19: Función de equidad nominal F_i, caso 3.

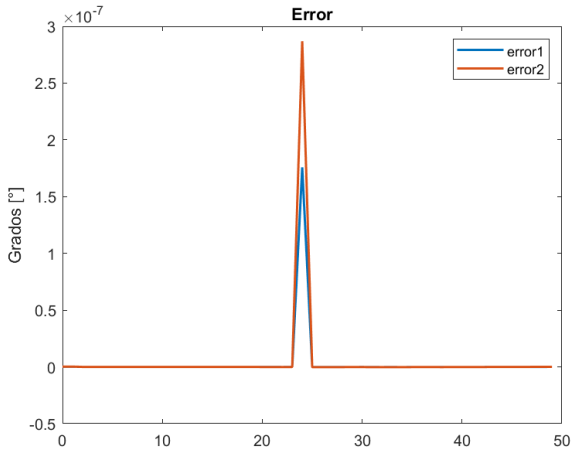

Figura 20: Señal de error giroscopio, caso 3.

8. Conclusiones

Se presentó un modelo para *RM* descrito por ecuaciones en diferencias, las cuales toman encuentra la distribución de recursos globales y locales, así como el comportamiento del *CBS*. Se diseñó un sistema difuso con el fin de obtener una distribución de recursos equitativa y realizar el rellenado del presupuesto según los recursos disponibles. A partir de esta descripción, se pudo diseñar una ley de control difuso para buscar convergencia a distribuciones equitativas, cuya convergencia se probó a partir de un análisis por linealización. Se presentaron resultados de simulaciones que confirman la viabilidad del esquema propuesto y enfatizan la necesidad de contar con los recursos mínimos necesarios para su correcta ejecución.

Agradecimientos

Este trabajo ha sido realizado parcialmente gracias al apoyo del CONACYT BECA 597175, PAPIIT IT100320 y PAPIIT IN104516.

Referencias

Abeni, L., Buttazzo, G., Dec 1998. Integrating multimedia applications in hard real-time systems. In: Proceedings 19th IEEE Real-Time Systems Symposium (Cat. No.98CB36279). pp. 4–13.
DOI: 10.1109/REAL.1998.739726

Aparicio, Santos, J.-A., 2017. Diseño de un controlador difuso para compensar cargas de comunicación en tiempo real. Master's thesis, Universidad Nacional Autónoma de México, México.

Bini, E., Buttazzo, G., Eker, J., Schorr, S., Guerra, R., Fohler, G., Arzen, K. E., Romero, V., Scordino, C., May 2011. Resource management on multicore systems: The actors approach. IEEE Micro 31 (3), 72–81.
DOI: 10.1109/MM.2011.1

Boutalis, Y., Theodoridis, D., Kottas, T., Christodoulou, M. A., 2014. System Identification and Adaptive Control: Theory and Applications of the Neuro-fuzzy and Fuzzy Cognitive Network Models. Springer.

Buttazzo, G. C., 2011. Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications, 3rd Edition. Springer Publishing Company, Incorporated.

Byeong Gi, L., Daeyoung, P., Hanbyul, S., 2009. Wireless Communications Resource Managemen. John Wiley and Sons.

Chasparis, G., Maggio, M., Arzen, K. E., Bini, E., June 2013. Distributed management of cpu resources for time-sensitive applications. In: 2013 American Control Conference. pp. 5305–5312.
DOI: 10.1109/ACC.2013.6580666

Chasparis, G. C., Maggio, M., Bini, E., Arzen, K.-E., 2016. Desing and implementation of distributed resource management for time-sensitive applications. Automatica 64, 44 – 53.
DOI: https://doi.org/10.1016/j.automatica.2015.09.015

Clark, R. K., 1990. Scheduling dependent real-time activities. Ph.D. thesis, USA, aAI9107552.

Ganz, A., Ganz, Z., Wongthavarawat, K., 2003. Multimedia Wireless Networks: Technologies, Standards and QoS. Pearson Education.

Horn, W., 1974. Some simple scheduling algorithms. Naval Research Logistics Quarterly 21 (1), 177–185.
DOI: 10.1002/nav.3800210113

IEEE, 1990. IEEE Standard Glossary of Software Engineering Terminology.
DOI: 10.1109/IEEESTD.1990.101064

Litoiu, M., Tadei, R., 2001. Fuzzy scheduling with application to real-time systems. Fuzzy Sets and Systems 121 (3), 523 – 535.
DOI: https://doi.org/10.1016/S0165-0114(99)00176-1

Mahmoud, M., of Engineering, I., Technology, 2013. Distributed Control and Filtering for Industrial Systems. Control, Robotics and Sensors. Institution of Engineering and Technology.
URL: https://books.google.com.mx/books?id=qWhWx2hRLYcC

Mok, A. K., Feng, X., May 2001. Resource partition for real-time systems. In: Proceedings Seventh IEEE Real-Time Technology and Applications Symposium. pp. 75–84.
DOI: 10.1109/RTTAS.2001.929867

Nesbit, K. J., Moreto, M., Cazorla, F. J., Ramirez, A., Valero, M., Smith, J. E., May 2008. Multicore resource management. IEEE Micro 28 (3), 6–16.
DOI: 10.1109/MM.2008.43

Quanser, 2012. USER MANUAL 3 DOF Gyroscope Experiment Set Up and Configuration. Quanser inc.

Robert H. Cannon, J., 2003. Dynamics Of Physical Systems. Dover Publications, INC.

Stankovic, J. A., 1988. Misconceptions about real-time computing: a serious problem for next-generation systems. Computer 21 (10), 10–19.

Subrata, R., Zomaya, A. Y., Landfeldt, B., Oct 2008. A cooperative game framework for qos guided job allocation schemes in grids. IEEE Transactions on Computers 57 (10), 1413–1422.
DOI: 10.1109/TC.2008.79

Tanaka, K., Ikeda, T., Wang, H. O., May 1998. Fuzzy regulators and fuzzy observers: relaxed stability conditions and lmi-based designs. IEEE Transactions on Fuzzy Systems 6 (2), 250–265.
DOI: 10.1109/91.669023