ScienceDirect
Disponible en www.sciencedirect.com
Revista Iberoamericana de Automática e Informática industrial 13 (2016) 196–206
www.elsevier.es/RIAI

Núcleo de Control: Controladores modulares en entornos distribuidos

Raúl Simarro*, José E. Simó, José Luis Navarro, José-Luis Poza-Luján, Juan-Luis Posadas-Yagüe

Instituto Universitario de Automática e Informática Industrial (ai2).
Universitat Politècnica de València, Camino de Vera, no14, 46022, Valencia, España

Resumen

En este artículo se describe una estrategia de control distribuida, utilizando elementos empotrados, mediante un *middleware* de control denominado núcleo de control, en el que se implementan controladores digitales de altas prestaciones, diseñados de forma modular, en sistemas con capacidad de cómputo limitada. Se presenta una metodología tanto para la obtención de una métrica que permita la comparación de los distintos modos de funcionamiento ante pérdidas de datos, como para la elección de los parámetros de los controladores a implementar en cada nodo del sistema distribuido de control. Se presentan varios modos de funcionamiento que permite adaptar el sistema desarrollado a distintas situaciones. El trabajo se completa con la implementación de la simulación del sistema distribuido y su prueba sobre un proceso real.

Palabras Clave:
Sistemas empotrados, control predictivo basado en modelo, control de sistemas distribuidos

1. Introducción

Los sistemas empotrados tienen un amplio rango de aplicabilidad en muchos sectores y su importancia crece continuamente. Uno de los campos de aplicación incluye la realización de tareas de control. La heterogeneidad de los sistemas actuales, formados por múltiples componentes de diferentes características conectados en red, sugiere el desarrollo de sistemas de control distribuido en los que las distintas funciones del control se implementen a diferentes niveles. El núcleo de control, asimilable al núcleo de un sistema operativo, se define como el código mínimo que debe ejecutarse en una aplicación de control para que el funcionamiento sea seguro, y es el encargado de la conexión entre los nodos del sistema distribuido.

Un sistema de control distribuido debe garantizar siempre la seguridad en el proceso que está regulando. La seguridad es un tema crucial en los sistemas empotrados de control (Li and Yao, 2003) (Albertos et al., 2005). Independientemente del número de variables controladas por el mismo procesador se debe garantizar la entrega de las acciones de control a los actuadores. La calidad de la señal entregada puede depender del nivel de procesamiento: los datos, algoritmos de cálculo y disponibilidad de recursos, entre otros, pero siempre se debe garantizar el funcionamiento seguro del sistema (Albertos et al., 2006)

* Autor en correspondencia.
Correo electrónico: rausifer@ai2.upv.es (Raúl Simarro),
URL: www.ai2.upv.es

El sistema de control debe hacer frente a problemas como pérdida de datos, retardos o tiempo excesivo de cálculo, consiguiendo una degradación admisible de prestaciones. Los controladores desarrollados deben permitir la adaptación de los mismos a las condiciones de trabajo, utilizando los recursos adecuados en cada momento, para no tener pérdidas significativas de prestaciones en el logro de los objetivos a conseguir.

El marco de aplicación de esta estructura de control son los sistemas ciber-físicos en los que las restricciones de recursos y los requisitos de control son fuertes (Kyoung-Dae and Kumar, 2012).

En (Rajkumar et al., 2010) se define a un sistema ciber-físico como un sistema que integra computación, capacidad de almacenamiento, seguimiento y control de entidades en el mundo físico, que operen de forma independiente, segura, fiable, eficiente, flexible y robusta. Es decir, se trata de sistemas de control con restricciones de tiempo real y espaciales actuando sobre un entorno físico, debiendo integrar los conceptos generales de control, computación y comunicación, con las características dinámicas de los sistemas físicos y de ingeniería.

El sistema de control distribuido presentado en este trabajo ha sido probado mediante una herramienta de simulación y análisis, creada a tal efecto, en procesos con distinta dinámica. Además, también se ha evaluado el sistema distribuido de control sobre varios procesos reales, lo que ha permitido la validación de la herramienta de simulación así como comprobar las prestaciones conseguidas ante problemas en las comunicaciones.

http://dx.doi.org/10.1016/j.riai.2015.11.005

2. Núcleo de control distribuido

El núcleo de control es asimilable al núcleo de un sistema operativo, y se ha desarrollado en los proyectos de investigación (KERTROL, 2006-2008), (SIDIRELI, 2009-2011) y (COBAMI, 2012-2014) en los que han trabajado los autores. La misión principal del núcleo de control es garantizar un funcionamiento seguro del proceso que se está regulando para el cumplimiento de las especificaciones deseadas. Así, el núcleo de control tiene que tomar las decisiones, en última instancia, que permitan llevar a cabo esta tarea de la forma más eficaz posible. Además, debido a su implementación en sistemas empotrados, el código empleado en este núcleo debe ser mínimo.

El núcleo de control tiene un carácter modular, de tal forma que, en función de la disponibilidad de recursos y de las tareas encomendadas a un sistema empotrado conectado en red, se estructura en niveles que garantizan la operación segura del sistema.

La implementación del núcleo de control se realiza en el entorno del núcleo del Sistema Operativo (SO), aunque no tiene porqué formar parte de éste, ofreciendo opciones compatibles con los distintos niveles de ejecución del SO. En concreto, el núcleo de control se materializa en la forma de un *middleware* sobre el que se ejecuta la aplicación de control, que está compuesta por diferentes actividades computacionales que presentan diferentes requisitos para su correcta ejecución. El *middleware* gestiona la ejecución de las actividades y ofrece los componentes básicos necesarios para el diseño de aplicaciones de control (reguladores, conmutadores, muestreadores, actuadores, retenedores,…), además incorpora servicios de evaluación de la "calidad del control" que permiten la detección de situaciones de excepción y la definición de acciones asociadas a estas situaciones.

El *middleware* ofrece un conjunto de servicios que hacen posible el funcionamiento de aplicaciones distribuidas sobre plataformas heterogéneas. Funciona como una capa de abstracción de software distribuida, que se sitúa entre las capas de aplicaciones y las capas inferiores (sistema operativo y red). El *middleware* nos abstrae de la complejidad y heterogeneidad de las redes de comunicaciones subyacentes, así como de los sistemas operativos y lenguajes de programación, proporcionando una interfaz de programación de aplicaciones (API) para la fácil programación y manejo de aplicaciones distribuidas. Dependiendo del problema a resolver y de las funciones necesarias, serán útiles diferentes tipos de servicios de *middleware*.

El *middleware* permite una adecuada separación entre el SO, si lo hubiese, y la aplicación de control, brindando un diseño rápido de aplicaciones y un desarrollo transparente (Albertos et al., 2006), (Crespo et al. 2006).

Una arquitectura software basada en componentes tiene varias ventajas (Baliga et al., 2005). Dado que los componentes están bien definidos pueden ser reemplazados sin afectar al resto del sistema. Una arquitectura de componentes también permite que cada uno de ellos pueda desarrollarse por separado y que sean fácilmente integrados más tarde, característica muy importante para el desarrollo de grandes sistemas. Además, este tipo de arquitectura promueve la reutilización de software, ya que un componente bien diseñado para un algoritmo de control, probado para un sistema, puede ser trasladado a otro sistema similar.

Se ha realizado la extensión del concepto de núcleo de control, en el sentido de considerar que la aplicación de control se ejecuta en un sistema distribuido interconectado mediante un sistema de comunicaciones. Esto ha llevado a la necesidad de considerar el concepto de "recurso limitado", no sólo en el caso de uso de la unidad central de procesamiento (CPU) por parte de las actividades de control, sino en el uso de los canales de comunicación que distribuyen la información. Diferentes configuraciones de ejecución de las actividades de control en los diferentes nodos que componen el sistema distribuido dan lugar a diferentes niveles de uso de los recursos involucrados, así como a diferentes niveles de calidad de control. Adicionalmente, cada configuración está asociada a un consumo energético por parte de los procesadores y las comunicaciones. Además de los problemas asociados a la distribución "optima" de actividades, en este contexto, se han desarrollado técnicas de control que permiten manejar situaciones en las que se producen fallos o retrasos inesperados en las comunicaciones de forma que la degradación en la calidad del control sea la menor posible.

2.1. Middleware del núcleo de control

El *middleware* del núcleo de control se ha desarrollado dentro de diversos proyectos de investigación por miembros del Instituto de Automática e Informática Industrial (ai2), para dar solución a la implementación de estrategias de control en distintas plataformas. La aproximación a este uso de *middleware* es similar a la seguida en otros dominios diferentes en los que se aumenta la lógica de un sistema distribuido, no olvidando la capacidad de comunicación inherente, con módulos para soportar la composición de aplicaciones distribuidas de tiempo real en tiempo acotado (García Valls et al., 2013).

Debido a la existencia de una gran variedad de sistemas empotrados con diversas características como: capacidades desarrolladas, tiempos de cómputo, interacción con el entorno, etc., es necesario establecer diversos niveles de servicios *middleware* para ser implementados en función de la restricción de recursos prevista por cada sistema. En (Coronel et al., 2008) se proponen dos modelos de *middleware* del núcleo de control compuestos por diversos grupos de servicios: un *middleware* completo (*Full-Middleware*) y un *middleware* reducido (*Tiny-Middleware*).

La versión completa implementa todas las funcionalidades del *middleware* de control, ofreciendo interfaces y mecanismos para la gestión de bucles complejos de control. Gestiona, organiza e interconecta toda la red distribuida de control.

El *middleware* completo se utiliza en nodos con procesadores empotrados potentes, con sistemas operativos de tiempo real y con altas capacidades de E/S y red.

Los principales componentes y servicios de la versión completa son (Muñoz et al., 2013):

- Gestor de la aplicación: Se encarga del acceso de la aplicación de control a los servicios ofrecidos por el *middleware*.
- Gestor de calidad de servicio (QoS): Negocia y gestiona la ejecución de las tareas del *middleware* y las aplicaciones del sistema para cumplir determinados niveles de rendimiento.
- Gestor de delegación: Coordina la delegación de código entre los distintos nodos de cómputo del sistema distribuido, en función de las peticiones de las aplicaciones, la disponibilidad de recursos locales, la distribución física de sensores y/o actuadores, etc.

- Gestor de datos: Este componente utiliza un modelo de intercambio céntrico de datos DCPS (*Data C entric Publish/Subscribe*), que es accesible por las aplicaciones interesadas.
- Data: Es el objeto que es generado por las aplicaciones publicadoras y consumido por las subscriptoras, en el espacio global de datos.
- Gestor de recursos: Este componente, en coordinación con el gestor de QoS, se encarga de administrar el uso de los recursos físicos de los sistemas de cómputo tales como memoria, CPU, ancho de banda, consumo de energía, etc.
- Gestor de red: Administra el hardware de red, accediendo a los controladores ofrecidos por el sistema operativo.
- Gestor de eventos de control: Informa al nivel de aplicación acerca de los eventos que ocurren dentro del *middleware* de control como pérdida de plazos de ejecución y errores de ejecución, entre otros.
- Servicio de actuación: Es el componente que se encuentra físicamente conectado al proceso a controlar y cuya misión es la entrega de las señales de control en cada periodo de actuación.
- Servicio sensorial: Este componente realiza el intercambio de información entre el proceso y los controladores involucrados. Con la información recibida en este componente el controlador debe elaborar las señales de control.

La versión reducida del *middleware* de control se comunica con la versión completa mediante interfaces básicas para la gestión de sensores, actuadores y código de controladores. Además, incluye interfaces de red para la interconexión a sensores y actuadores, y para la descarga de código de controladores desde nodos con la versión completa. El nivel operativo está formado por un conjunto de API's de control hardware con planificación cíclica.

Esta versión está dirigida a nodos con procesadores con capacidad de cómputo y red limitada, pero con características completas de E/S, como por ejemplo PIC (controlador de interfaz periférico), DSP (Procesador Digital de Señales), etc.

La versión reducida implementa una versión básica de los siguientes componentes y servicios, que desde el punto de vista conceptual y lógico tienen las mismas características descritas anteriormente: Gestión de datos y controladores, Servicio de delegación, Servicio de red, Servicio de actuación y el Servicio sensorial.

Un servicio adicional de este componente es el siguiente:

- Servicio de eventos y monitorización: Es el encargado por una parte de monitorizar el control efectuado por el nodo de recursos limitados. Además, tiene encomendadas tareas de administración del *middleware*, tales como por ejemplo el modo de funcionamiento del núcleo de control o el estado del nodo.

2.2. Estructura del sistema distribuido de control

La figura 1 muestra la estructura propuesta del sistema distribuido de control, formada por nodos supervisores y nodos locales (Simarro et al., 2008).

Los nodos supervisores son potentes computadores ejecutando un sistema operativo de tiempo real (RTOS) y con completas capacidades de red y entrada/salida, en concreto, para las pruebas experimentales se ha utilizado una versión modificada de RTLinux 3.2rc1 para procesadores XScale (Coronel 2007).

Las aplicaciones de control de alto nivel se ejecutan en los nodos supervisores, que implementan un *middleware* completo del núcleo de control con las características descritas anteriormente: abstracciones y funcionalidades relacionadas con la ejecución de tareas de control en tiempo real, acceso a los sensores y actuadores, y gestión de comunicaciones.

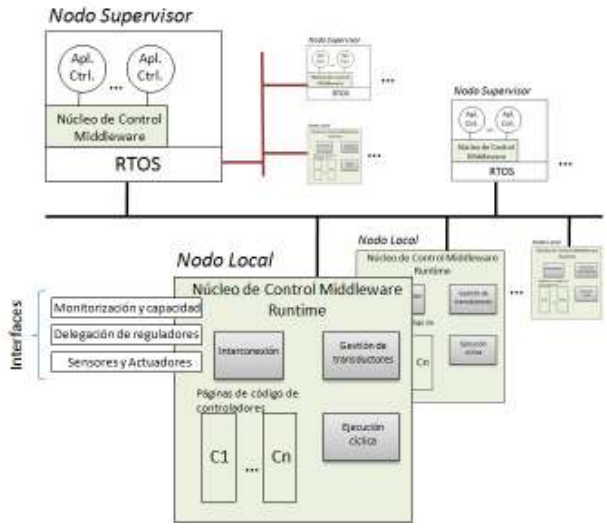

Figura 1: Arquitectura para el modelo de control en un sistema distribuido.

Los nodos locales son procesadores SoC (*System o n C hip*) pequeños y de bajo consumo con capacidades limitadas de redes y computación, pero características completas de entrada/salida.

Los nodos locales son una solución económica de disponer de capacidad de cómputo cerca de cada actuador. Esta característica es obligatoria para reducir el indeterminismo en el tiempo de envío de las acciones de control al proceso, realizando la entrega en el instante adecuado, aunque debido a las limitaciones en el tiempo de cómputo inherentes a este nodo se deben realizar los mínimos cálculos posibles para que se asegure el cumplimiento de este objetivo. Los nodos locales ejecutan un *middleware* reducido del núcleo de control, con las características descritas anteriormente en el apartado 2.1.

El objetivo fundamental de los nodos locales es asegurar que siempre exista una acción de control para mandar al proceso. Esta señal de control puede ser una acción de control segura (desconectar, abrir, cerrar, mantenerse sin cambios, etc.), el resultado de un cálculo sencillo mediante un PID (Controlador Proporcional-Integral-Derivativo) localmente implementado en el nodo, que garantice como mínimo la estabilidad del sistema, o puede ser la señal calculada y enviada desde el nodo supervisor (Simarro et al., 2013).

2.3. Modos de funcionamiento

Se propone la implementación de varios modos de funcionamiento, que permitan al sistema adaptarse a las características del mismo y cumplan las especificaciones del control con una degradación en las prestaciones del sistema adecuada. Cuando el canal de comunicaciones no es de uso

exclusivo se producen una serie de problemas que el algoritmo de control deberá tener en cuenta, como pueden ser: la sincronización entre los nodos, el tratamiento de los retardos y desfases, y el tratamiento de distintos periodos de actuación entre los nodos.

Estos modos de funcionamiento se implementan en la aplicación de control, y por lo tanto hacen uso de las API que proporciona el middleware del núcleo de control presentado anteriormente. Se puede encontrar más información de las interfaces proporcionadas por el middleware y sus características en (Beltrán et al., 2014) y (Poza et al., 2009).

La estrategia de control que se describe a continuación es válida para procesos con una entrada y una salida, procesos SISO, utilizando un nodo supervisor y un nodo local para la estructura de control, aunque las estrategias de control presentadas se pueden ampliar a un sistema con un nodo supervisor y varios nodos locales, como el que se muestra en la figura 1, con relativa facilidad.

La figura 2 muestra la estructura del sistema distribuido formado por 2 nodos, en la que se muestra el intercambio de datos entre cada uno de ellos. A continuación se describen cada uno de los modos de funcionamiento desarrollados, en los que se especifica el intercambio de datos para cada uno de ellos.

Figura 2: Estructura del sistema distribuido de control con 2 nodos

Modo 1. Control jerarquizado. El nodo supervisor envía las referencias y el local las sigue mediante un regulador seguidor sencillo. Este modo de funcionamiento es similar a un control jerarquizado con un canal de comunicaciones compartido.

En (1) se presentan las señales que el nodo supervisor envía hacia el local y viceversa.

$$\begin{aligned} \delta sl(k) &\equiv R_{ef}(k) \\ \theta ls(k) &\equiv y(k) \end{aligned} \tag{1}$$

La labor del nodo supervisor es la coordinación del comportamiento de la salida del proceso, indicando al nodo local el momento en el que debe cambiar las consignas. El nodo local realizaría las tareas de lectura de la variable bajo control y el envío hacia el control supervisor, ya que, el nodo supervisor debe ser informado de la situación de la variable bajo control.

En caso de pérdida de datos el control local siempre puede utilizar la última referencia para mantenerse en ella, o proponer la referencia que le lleve a una situación segura.

Modo 2. Control basado en red. El nodo supervisor envía las acciones de control al nodo local y éste las aplica sobre el proceso. Este modo de funcionamiento es similar al control digital directo con un canal de comunicaciones compartido, es lo que se denomina un control basado en red.

En (2) se presentan las señales intercambiadas por cada nodo.

$$\begin{aligned} &\delta sl(k) \equiv u(k) \\ &\theta ls(k) \equiv \begin{Bmatrix} y(k) \\ Y_{anterior} = \{y(k-1),...,y(k-n)\} \\ U_{anterior} = \{u(k-1),...,u(k-n)\} \end{Bmatrix} \end{aligned} \tag{2}$$

En este modo de funcionamiento el nodo local carece de un sistema de control, ya que es el nodo supervisor el que lo implementa. En caso de pérdida de datos, o de problemas en la recepción de la acción de control, el nodo local no recibiría la información actualizada, y por lo tanto debería aplicar la acción de control anterior (o incluso realizar una secuencia de seguridad con paro). Por este motivo el nodo local envía hacia el nodo supervisor las salidas y acciones de control aplicadas en instantes anteriores, para que así el control supervisor calcule la nueva acción de control con los datos actualizados.

Modo 3. Envío de las acciones de control postuladas. En el nodo supervisor se ejecuta un control de alto nivel, como por ejemplo un regulador predictivo, capaz de calcular las trayectorias de las acciones de control postuladas en un horizonte de predicción. El nodo supervisor envía las trayectorias de las acciones de control hacia el nodo local en cada instante de muestreo. En este modo, en caso de pérdida de datos se podría utilizar las acciones de control postuladas, aunque se estaría realizando un control en bucle abierto.

En (3) se muestran las señales intercambiadas por cada nodo.

$$\begin{aligned} &\delta sl(k) \equiv U = \{u(k|k),...u(k+Nu-1|k)\} \\ &\theta ls(k) \equiv \begin{Bmatrix} y(k) \\ Y_{anterior} = \{y(k-1),...,y(k-n)\} \\ U_{anterior} = \{u(k-1),...,u(k-n)\} \end{Bmatrix} \end{aligned} \tag{3}$$

Modo 4. Envío de las acciones y las salidas postuladas. La diferencia de este modo con el anterior es que el nodo supervisor envía las trayectorias de las acciones de control junto con las salidas postuladas (4). En caso de pérdida de datos el control local puede verificar si la salida real se corresponde con la postulada por el control supervisor y, en caso de discrepancia, realizar una compensación de la acción de control mediante el error cometido en la salida.

$$\begin{aligned} &\delta sl(k) \equiv \begin{Bmatrix} Y = \{\hat{y}(k+1|k),...,\hat{y}(k+N|k)\} \\ U = \{u(k|k),...u(k+Nu-1|k)\} \end{Bmatrix} \\ &\theta ls(k) \equiv \begin{Bmatrix} y(k) \\ Y_{anterior} = \{y(k-1),...,y(k-n)\} \\ U_{anterior} = \{u(k-1),...,u(k-n)\} \end{Bmatrix} \end{aligned} \tag{4}$$

El método utilizado para realizar la compensación de la acción de control se explica en el apartado 2.4.

Modo 5. Envío de las acciones y las salidas postuladas en los cambios de referencia. Este modo de funcionamiento es similar al anterior, pero en este caso el nodo supervisor envía las trayectorias de las acciones de control y de las salidas postuladas en los cambios de referencia. Es decir, el nodo supervisor no realizará ningún cálculo ni envío mientras el nodo local disponga de información suficiente, por lo que el nodo local debe utilizar estas trayectorias, aplicando la compensación de la acción de control, para seguir con la trayectoria nominal de la salida hasta que reciba nuevos datos, consecuencia de una actualización por parte del control supervisor. El intercambio de señales entre los nodos se corresponde con (4).

El cálculo de trayectoria nominal realizada por el control supervisor debería llevar en última instancia al sistema a una situación estable, para que en caso de perder los datos entre el nodo supervisor y el nodo local éste pueda llevar a la salida del sistema a ese estado realizando pequeños cambios en la acción de control para contrarrestar el efecto de perturbaciones.

Una característica de este modo de funcionamiento es que, el nodo local, puede enviar un evento al nodo supervisor para que, en caso de error significativo entre la salida nominal y la salida actual del proceso, se mande al control supervisor recalcular las trayectorias de las acciones de control y de la salida.

2.4. Compensación de la acción de control

En los modos de funcionamiento 4 y 5 se utiliza la información de la discrepancia, entre la salida prevista y la real, para compensar la acción de control, y evitar, en caso de pérdida de datos, la utilización directa por parte del nodo local de las acciones de control postuladas, ya que se estaría utilizando una estrategia de control en bucle abierto.

Al utilizar la información postulada, calculada en el instante k para el instante i, dentro del horizonte de predicción, se define el error en la predicción de la salida como (5):

$$ey(k+i\,|\,k) = \hat{y}(k+i\,|\,k) - y_{real}(k+i) \quad i = 1,...,N \qquad (5)$$

La estrategia utilizada en el control local a partir del error en la salida y de la información disponible será (6):

If i<Nu-1, then

$$u = u(k+i\,|\,k) + K_{gain} \cdot ey(k+i\,|\,k)$$

If (i≥Nu-1)&(i<N), then

$$u = u(k+N_u-1\,|\,k) + K_{gain} \cdot ey(k+i\,|\,k) \qquad (6)$$

If (i≥N), then

$$u = u(k+N_u-1\,|\,k) + K_{gain} \cdot (\hat{y}(k+N\,|\,k) - y_{real}(k+i))$$

donde:

- $\hat{y}(k+i|k)$ es la salida postulada por el control predictivo para el instante de tiempo $k+i$, calculada con la información hasta el instante k
- $y_{real}(k+i)$ es la salida real medida en el proceso real en el instante $k+i$
- K_{gain} es la ganancia que debe aplicar el compensador local para minimizar el error en la salida

En la compensación de la acción de control se ha optado por una estrategia de control sencilla en los nodos locales, ya que son los que más restricciones tienen desde el punto de vista computacional, pero también se podría utilizar en el nodo local un regulador PID o bien un control de mínima varianza. La compensación de la acción de control propuesta utilizando solo una ganancia tiene las siguientes propiedades (Simarro 2011):

Propiedad 1. El nodo local implementa la estrategia descrita en (6), y la única información que necesita cuando aplica la técnica de la compensación de la acción de control es la ganancia.

Propiedad 2. La compensación de la acción de control se realiza para anular el efecto del ruido de la salida sobre el proceso y reducir el efecto de las perturbaciones y errores de modelado. La ganancia a aplicar debe ser relativamente pequeña, ya que lo que se busca es ajustar la acción de control predicha a las condiciones reales.

Propiedad 3. La compensación de la acción de control puede utilizarse tanto en los casos esporádicos de pérdida de datos o tiempo excesivo de espera, como en la utilización del envío de las trayectorias de la acción de control y de la salida sólo en los cambios de referencia. En este último caso la compensación de la acción de control se utilizará en el nodo local hasta el envío de las siguientes trayectorias por parte del nodo supervisor.

Propiedad 4. En procesos no estables la compensación de la acción de control actúa como un lazo de control en bucle cerrado cuando se pierde la conexión con el nodo supervisor. Las pruebas, tanto simuladas como reales, sugieren que, en los sistemas inestables, se puede ganar tiempo antes de llegar a una región de recuperación imposible, por lo que al utilizar esta técnica se pueden asumir retrasos o pérdidas de datos mayores que si no se hiciese así.

2.5. Elección de los parámetros de control en los nodos del sistema distribuido

En el caso de utilización de los modos de funcionamiento 3, 4 y 5, en los que el nodo supervisor envía las trayectorias de las acciones de control y la salida (en el caso de los modos 4 y 5), es necesario diseñar los reguladores de forma que se tenga en cuenta que el nodo local puede utilizar la información postulada en caso de pérdida de datos.

Diversos trabajos proponen controladores MPC (Control Predictivo Basado en Modelo) que permiten la reducción del uso de las comunicaciones mediante un muestreo no periódico (Berglind et al., 2012), (Eqtami et al., 2011) y (Heemels et al., 2012), o bien la utilización de un control descentralizado y distribuido (Camponagara et al., 2012), (Farokhi et al., 2014), (Giselsson and Rantzer, 2010) y (Giselsson and Rantzer, 2014). Sin embargo, en estos esquemas de control es necesaria la utilización de procesadores de altas prestaciones. Con la propuesta de los modos de funcionamiento anteriores se permite la utilización de un control local con bajos recursos de cómputo, mediante la utilización de la compensación de la acción de control, y la implementación de controles sencillos en última instancia para asegurar el envío de una acción de control.

Los valores más adecuados de los parámetros de cada uno de los nodos del sistema distribuido de control cambian en función del proceso a controlar, las condiciones del entorno (perturbaciones, discrepancias, etc.) y las características del canal de comunicaciones. Así, es necesario analizar estas características

para encontrar los parámetros más adecuados, utilizando un método heurístico, que permita minimizar los errores al utilizar las trayectorias de la salida y control postuladas. Para ello se cuenta con la ayuda de una herramienta de simulación y análisis del sistema distribuido creada a tal efecto (Simarro 2011). En el apartado 4 se presenta esta herramienta.

La **elección de los parámetros de control en el nodo supervisor**, considerando la utilización de un control predictivo MPC, es la siguiente:

- El horizonte de predicción N debe contener a la dinámica del sistema, por lo que ese será el mínimo número de instantes que debe tener.
- El horizonte de control Nu no debe cambiar bruscamente, y al utilizar las acciones postuladas la salida debe parecerse lo máximo posible a la salida del sistema al utilizar la técnica del horizonte deslizante. Es decir, que al aumentar el horizonte de control se permite que la acción de control se realice durante más instantes, pero aumenta de forma significativa el tiempo invertido por el nodo supervisor en el cálculo de los vectores correspondientes a las acciones de control y las salidas en cada instante. Sin embargo, esto no es problema teniendo en cuenta las características de ese nodo, aunque siempre es deseable que dicho tiempo sea el menor posible.
- Para cada horizonte de predicción N se utiliza un método heurístico en el que se evalúan todos los posibles horizontes de control (desde Nu=2 hasta Nu=N), con las condiciones de perturbaciones y canal de comunicaciones que se tienen en el sistema real, para el caso más desfavorable, que es cuando se utilizan las trayectorias postuladas en un cambio de referencia, sin utilizar ningún método de compensación. La combinación de N y Nu que consiga un error más pequeño será la más adecuada. La herramienta de simulación permite realizar el método de búsqueda de la mejor combinación de forma automática.

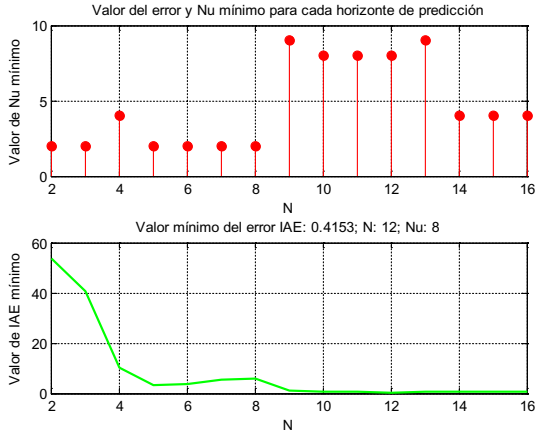

Figura 3. Mínimo IAE para cada N y Nu que hay que aplicar para conseguirlo

La figura 3 muestra, al aplicar la estrategia sobre un proceso de 2º orden subamortiguado, que se presenta en el apartado 5, como varía el error cometido en la salida, entre el valor predicho y el real cuando se utiliza las acciones postuladas, para los distintos horizontes de predicción (N) y cuál es el valor de Nu que consigue ese error mínimo. De todo ese conjunto de valores se puede obtener aquella combinación que consigue el mínimo error de todos. El índice utilizado para el cálculo del error ha sido el absoluto (IAE), en el apartado 3 se presentan los índices de error utilizados para la estimación de la calidad de la respuesta de control.

La aplicación de la compensación de la acción de control sirve para mejorar la respuesta del sistema cuando se utilizan las acciones de control postuladas, de forma que no se realice un control en bucle abierto. La elección de la ganancia que multiplica al error en la salida es muy importante, ya que dependiendo de su valor puede hacer que el comportamiento mejore o empeore comparado con la aplicación directa de las acciones postuladas.

En la **elección de los parámetros del control en el nodo local** se sigue el siguiente método:

- En el nodo local se utiliza la información proporcionada por el nodo supervisor, y en algunos casos se puede utilizar la información postulada con la compensación de la acción de control. Por este motivo lo que se plantea es un método heurístico, utilizando las condiciones de perturbaciones y canal de comunicaciones, para encontrar el valor de la ganancia de compensación, utilizando los parámetros del nodo supervisor encontrado con el método anterior, cuando se utiliza la información postulada para el caso más desfavorable, es decir, cuando el nodo local debe utilizar las trayectorias postuladas en un cambio de referencia.

La figura 4 muestra los errores cometidos por la salida real, con respecto a la trayectoria nominal, para distintas K cuando se utiliza la predicción con la compensación de la acción de control, para el caso de un proceso de 2º orden subamortiguado, cuyo modelo se presenta en la sección 5.

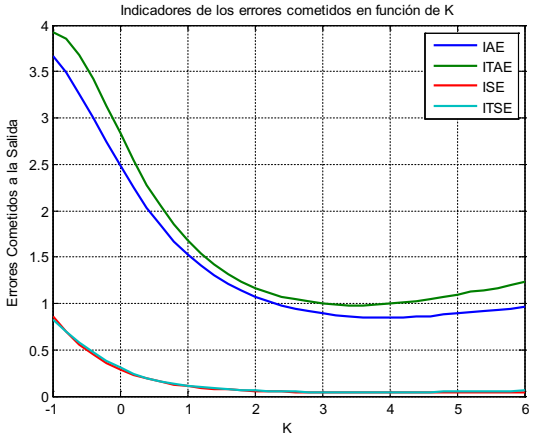

Figura 4. Errores en función de K aplicados a un proceso de 2º orden

Cualquiera de los indicadores de error (Dorf et al., 1995), tanto los absolutos (IAE, ITAE) como los cuadráticos (ISE, ITSE) presentan la misma tendencia.

Para el problema estudiado, la figura 4 muestra como los valores adecuados de K para el proceso y las condiciones del entorno tiene un mínimo error alrededor de K=4.

3. Estimación de la calidad de la respuesta de control

El objetivo de la estrategia de control desarrollada es que la salida del proceso real coincida con la trayectoria propuesta por el control supervisor. Esta trayectoria es tomada por el nodo local como una trayectoria nominal y que por lo tanto se considera como la salida deseada.

El error cometido en el instante k será (7):

$$Error(k) = y_{des}(k) - y_{act}(k) \quad (7)$$

Siendo y_{des} la salida deseada e y_{act} la salida actual

El índice de error utilizado es el del error absoluto IAE (Dorf et al., 1995). La expresión de este índice en un sistema discreto es (8):

$$IAE = \sum_{i=0}^{n} | y_{des}(k) - y_{act}(k) | \quad (8)$$

Siendo "n" el número de muestras consideradas para el cálculo del error.

La reducción en las prestaciones que sufre un sistema distribuido cuando se produce la pérdida de datos depende del instante en el que se produce la pérdida, y esto puede variar de un proceso a otro. Además, esta reducción de prestaciones es muy dependiente de la pérdida de datos producidas anteriormente, y por lo tanto de la situación en la que se encuentre la salida del proceso.

Para evaluar cómo afectan las distintas pérdidas de datos a las prestaciones del sistema se propone el siguiente método:

- Someter al sistema a varias pruebas con el mismo porcentaje de pérdidas de datos, pero distribuidas de forma aleatoria a lo largo del tiempo de la prueba.
- Se tomará como indicador de la pérdida de prestaciones el error cometido en la salida del proceso en cada instante respecto de la salida del proceso sin pérdida de datos.
- El indicador que determina las prestaciones para cada porcentaje de pérdidas será tomar el mayor índice obtenido para el conjunto de pruebas con ese porcentaje de pérdidas. Este índice expresa el peor de los casos encontrado en cada porcentaje de pérdidas. En las pruebas realizadas también se ha representado el valor medio del error cometido para cada porcentaje de pérdidas, lo que permite mostrar si el peor caso es un valor puntual o hay una tendencia.
- Repetir las pruebas para evaluar la pérdida de prestaciones con cada porcentaje de pérdidas.
-

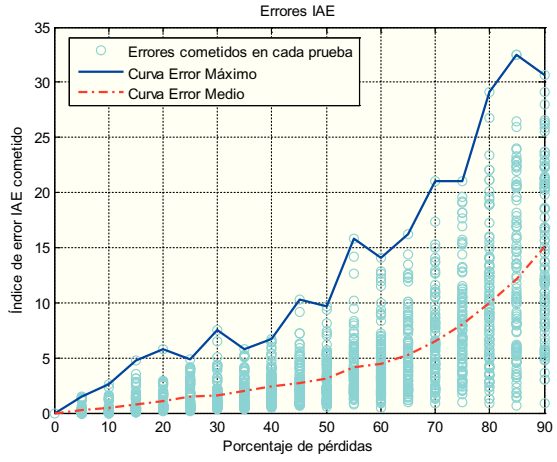

Figura 5: Curvas de error

En la figura 5 se muestran, para el caso del proceso de 2º orden que se presenta en la sección 5, las curvas de los errores IAE máximos y medios cometidos por cada porcentaje de pérdidas de datos, en el que se han realizado 100 pruebas con cada porcentaje de pérdida de datos distribuidas de forma aleatoria en el tiempo de la prueba.

4. Herramienta de asistencia al diseño del sistema distribuido de control

La herramienta de asistencia al diseño y simulación del sistema permite la obtención de los parámetros más apropiados de los controladores a implementar en los nodos del sistema distribuido, así como la simulación de distintas estrategias de control y modos de funcionamiento.

La herramienta está desarrollada en Matlab, lo que permite la utilización de funciones matemáticas, la simulación de estructuras de control con Simulink, la implementación de los algoritmos de control mediante funciones y las representaciones gráficas de una manera sencilla.

Figura 6: Bloques funcionales de la herramienta de simulación

La figura 6 muestra los bloques principales de la herramienta de simulación. El nodo supervisor implementa un control predictivo. En la estrategia de control propuesta, al necesitar las trayectorias obtenidas en cada iteración de las acciones de control y las salidas, se han tenido que desarrollar nuevas funciones de cálculo de los reguladores predictivos, ya que las existentes simplifican la solución, al utilizar la estrategia del horizonte deslizante.

En el nodo local se implementa la estrategia de actualización de los vectores internos a partir de la información recibida, teniendo en cuenta que puede haber discrepancias en los periodos de los nodos, desfases en la recepción de los datos, etc.

El código implementado en los bloques del nodo supervisor y local es similar al que debería implementarse en los nodos reales, por lo tanto, las pruebas llevadas a cabo por el simulador permiten, además de comprobar el funcionamiento, la creación del código necesario para su implementación real.

Para probar la ejecución en los nodos del sistema distribuido, los tiempos de cómputo, el canal de comunicaciones compartido, el ancho de banda, etc., se ha utilizado Truetime (Ohlin et al., 2007), que es una toolbox basada en Matlab/Simulink que permite simular el comportamiento de un núcleo de ejecución de tiempo real, así como el medio de comunicación utilizado para la transmisión de señales.

La figura 7 muestra la estructura del sistema distribuido de control con Simulink utilizado en la herramienta, en el que se simulan los nodos y el canal de comunicaciones mediante Truetime.

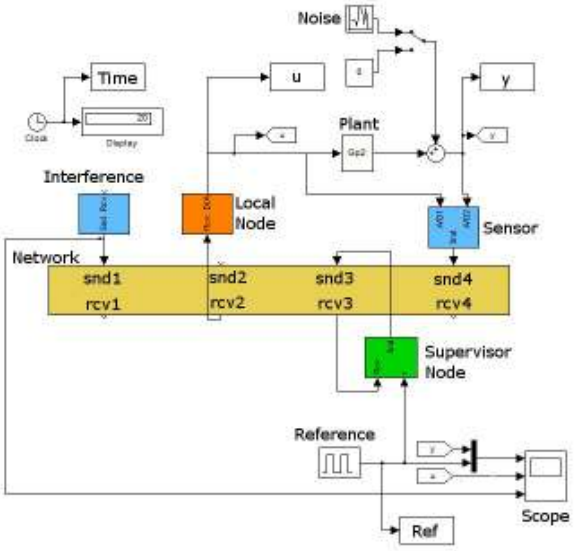

Figura 7: Estructura del sistema distribuido utilizando Truetime

5. Trabajo experimental

La estrategia de control propuesta se ha probado con procesos SISO con dinámicas diferentes. La elección de los distintos procesos se ha realizado teniendo en cuenta que pertenecen a procesos reales y que han sido documentados en algún trabajo previo. Las características de los procesos elegidos son similares a los modelos para la evaluación en el Benchmark 2010-2011 del Grupo Temático de Ingeniería de Control de CEA para la Evaluación de algoritmos de Auto-Ajuste de Controladores PID (Benchmark CEA 2010-2011).

Los procesos elegidos son: proceso de primer orden, de segundo orden, de fase no mínima, con retardo múltiplo del periodo de muestreo y proceso con efecto integral. En (Simarro 2011) se encuentran detalladas todas las pruebas y análisis de los procesos elegidos.

A continuación se muestran las pruebas, tanto simuladas como reales, donde se aplica la estrategia de control propuesta. Para la realización de estas pruebas se utiliza un proceso de segundo orden con respuesta subamortigada ante entrada escalón, que se corresponde con el circuito electrónico de la figura 8.

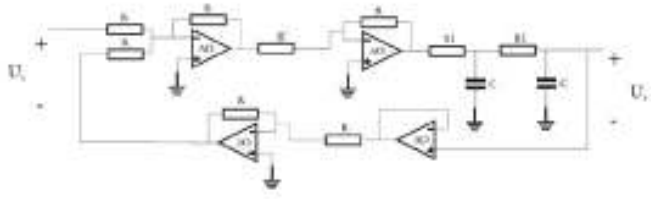

Figura 8: Circuito electrónico con respuesta subamortiguada ante un escalón

El modelo del proceso en forma de diagrama de bloques se muestra en la figura 9.

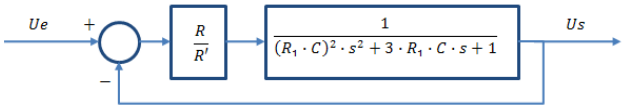

Figura 9: Diagrama de bloques del circuito electrónico utilizado

Los valores que toman los diferentes componentes en el circuito electrónico son los siguientes:

- R = 150 KΩ
- R'=22 KΩ
- R1=470 KΩ
- C=0.68 μF

Este proceso tiene la función de transferencia (9):

$$Gp(s) = \frac{Us(s)}{Ue(s)} = \frac{6.818}{0.1021 \cdot s^2 + 0.9588 \cdot s + 7.818} \qquad (9)$$

La respuesta del proceso real cuando se utiliza un control predictivo CRHPC (Control Predictivo Restringido de Horizonte Deslizante), utilizando un periodo de muestreo de 50 milisegundos y sin pérdida de datos se muestra en la figura 10.

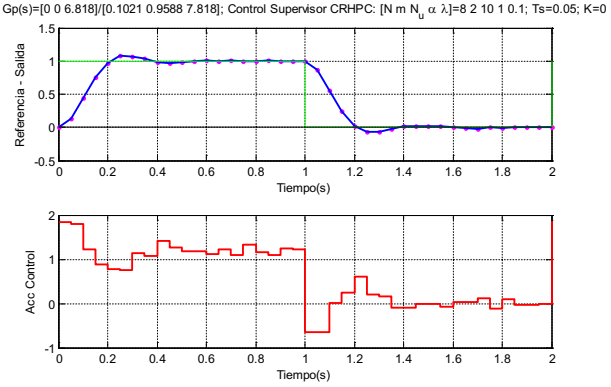

Figura 10: Respuesta del sistema distribuido sin pérdida de datos

Con este control se consigue un tiempo de establecimiento menor de 0.4 segundos con una sobreoscilación menor del 15%. En (Simarro 2011) se encuentra desarrollado el diseño de este controlador y su implementación

Previamente se realiza un análisis del sistema distribuido de control que permite determinar los parámetros más apropiados de los controladores de cada nodo (horizontes de control y predicción, restricción final, ganancia de compensación,…), con el método explicado en el apartado 2.5.

La simulación del sistema de control utilizando un canal de comunicaciones compartido en el que se provocan distintos porcentajes de pérdidas de datos, respecto del tiempo de la simulación, y en el que los instantes en los que se pierden las muestras son aleatorios se muestra en la figura 11. En las pruebas realizadas se comparan los modos de funcionamiento desde el 2 al 5, que son aquellos en los que las acciones de control son calculadas por el nodo supervisor mediante controladores de alto nivel.

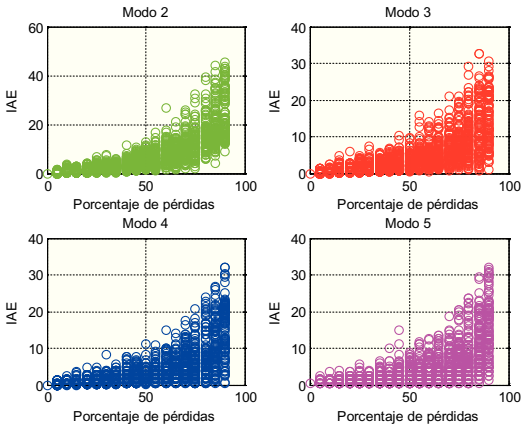

Figura 11: IAE de los distintos modos de funcionamiento en simulación

A continuación se representan sobre la misma gráfica los modos de funcionamiento probados con el error máximo obtenido por cada porcentaje de pérdida de datos, lo que permite considerar el peor caso, así como el valor medio, con el que se obtiene la tendencia de los errores.

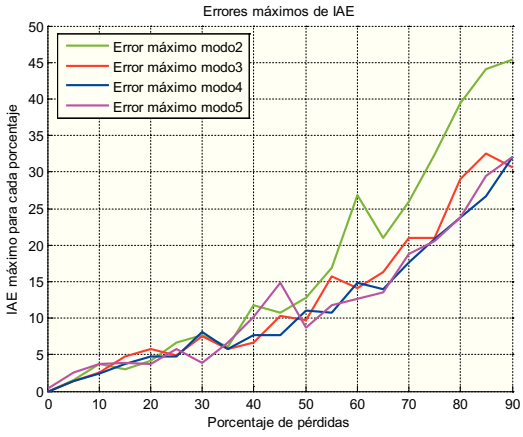

Figura 12: Curvas del error máximo por cada modo de funcionamiento

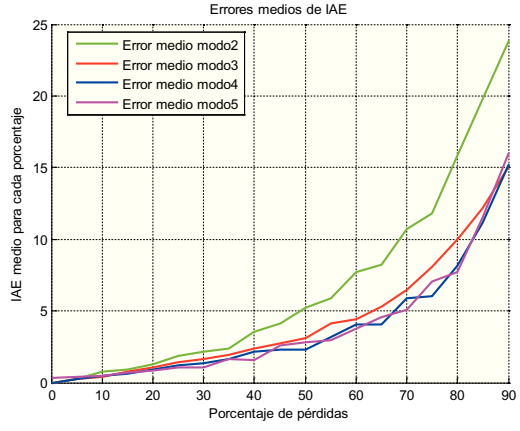

Figura 13: Curvas del error medio por cada modo de funcionamiento

Las figuras 12 y 13 muestran como los modos de funcionamiento que utilizan las salidas postuladas son los que obtienen unos índices de error más bajos en cada porcentaje de pérdidas, sobre todo cuando se pierden más del 40% de los datos. Además, los modos de funcionamiento 4 y 5, que son los que aplican la compensación de la acción de control con el error cometido en la salida, son los que obtienen unos índices de error más bajos.

Una vez analizado el comportamiento del sistema distribuido en simulación se procede a la implementación de la estrategia de control sobre el sistema real. El nodo supervisor, en el que se ejecuta un control predictivo, está implementado en un microprocesador XScale, mientras que el nodo local será implementado en un sistema empotrado que está formado por un dsPIC de Microchip y un sistema de comunicaciones CAN, que es el que permitirá el paso de información entre el nodo supervisor y el nodo local (Simarro 2011).

El nodo supervisor utiliza el sistema operativo de tiempo real RTLinux (Yodaiken y Barabanov 1997), que ofrece características de sistema de tiempo real estricto en un núcleo de tiempo real con multi-hilos, y que se puede utilizar como sistema operativo empotrado. Para ello es necesario hacer una portabilidad para adaptar el código fuente original de RTLinux a la arquitectura XScale, disponible en (Coronel 2007).

La figura 14 muestra la conexión utilizada entre los distintos elementos que forman parte del sistema distribuido en la prueba real. El ordenador es utilizado para monitorizar los datos y guardar la información de las pruebas, además, es el encargado de provocar las distintas situaciones de fallo.

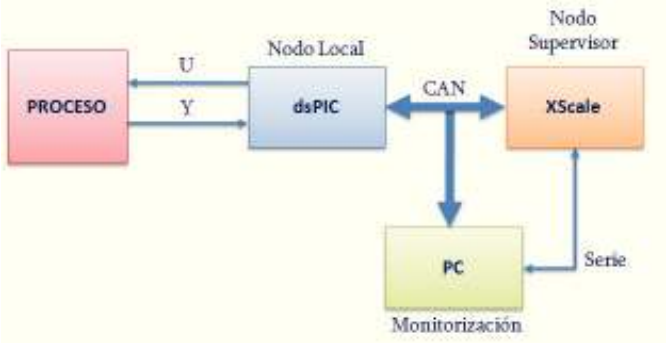

Figura 14: Estructura de control distribuida para la realización de la prueba

A continuación se muestran las pruebas realizadas sobre el proceso real del comportamiento de los distintos modos de funcionamiento cuando se producen pérdidas de datos. Para ello en el control supervisor se ha implementado un algoritmo que provoca las pérdidas de datos de forma aleatoria a lo largo de todo el tiempo de la simulación con distintos porcentajes, es decir, de una manera similar a la que se puso en práctica en simulación.

El middleware es el encargado de llevar a cabo la lectura de las variables de proceso mediante el servicio de datos y comunicarlas a los nodos. Ante una pérdida de datos el servicio del gestor de eventos informa al nodo local, que en función del modo de funcionamiento implementado enviará la acción de control adecuada al proceso.

Las figuras 16 y 17 muestran la comparación de los valores máximos y medios del IAE en cada porcentaje de pérdida de datos con los distintos modos de funcionamiento.

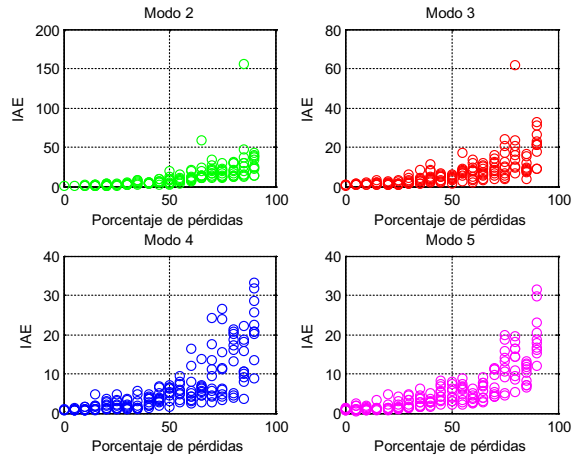

Figura 15: IAE de los distintos modos de funcionamiento en la prueba real

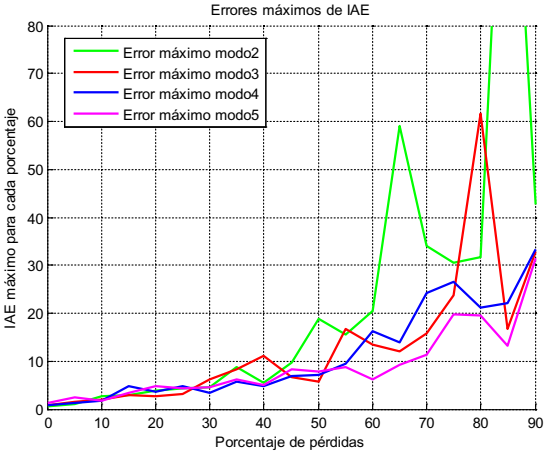

Figura 16: Índices de error máximos para cada modo en la prueba real

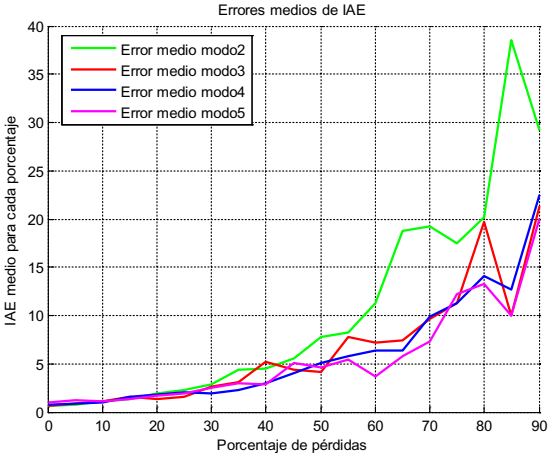

Figura 17: Índices de error medios para cada modo en la prueba real

La figura 18 muestra el comportamiento de la salida del sistema real ante el mismo patrón de pérdidas cuando se utilizan los distintos modos de funcionamiento. El porcentaje de pérdidas que provoca este patrón es de un 40%.

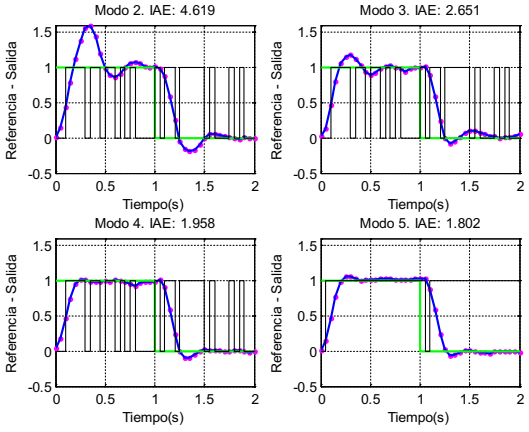

Figura 18: Respuesta real del proceso ante las mismas pérdidas de datos

La respuesta del proceso ante la misma pérdida de datos para los distintos modos de funcionamiento obtiene datos que son similares a los estudiados con las pérdidas porcentuales utilizando el peor caso. Los modos de funcionamiento que mejores resultados obtienen son aquellos que utilizan las acciones de control postuladas, y sobre todo los que además usan la técnica de la compensación de la acción de control. Las salidas de los procesos para estos casos son muy parecidas a las obtenidas en ausencia de pérdidas, obteniendo unas especificaciones de tiempo de establecimiento y sobreoscilación similares a las conseguidas sin pérdida de datos.

En el caso del modo de funcionamiento 5 la pérdida de datos afecta en aquellos instantes en que se actualizan las trayectorias nominales que envía el control supervisor al local. Como se muestra en la figura 18, se produce un retraso en el envío de las trayectorias al coincidir el cambio de referencia con la pérdida de un dato.

6. Conclusiones

La estrategia de control propuesta permite, al sistema de control, hacer frente a los problemas debidos a la utilización de un canal de comunicaciones compartido y las características de los nodos con poca capacidad de cómputo.

El middleware de control es el encargado de gestionar la adquisición de las señales e informar mediante un evento a los nodos en caso de pérdida de datos.

La utilización de las acciones de control y las trayectorias predichas mejora la respuesta del sistema ante pérdidas de datos, sobre todo cuando se utiliza la compensación de la acción de control.

La herramienta de asistencia al diseño permite la selección de los parámetros más apropiados a implementar en cada nodo del sistema distribuido.

La aplicación sobre ejemplos reales muestra un comportamiento muy bueno cuando se producen pérdidas de datos al utilizar la estrategia de control propuesta y los modos con compensación de la acción de control, permitiendo además validar las pruebas realizadas con el simulador.

English Summary

Control kernel: Modular controllers in distributed environments

Abstract

In this paper a distributed control strategy described using embedded items, using a control middleware called control kernel, in which high performance digital controllers designed in a modular way, on systems with limited computational capabilities are implemented. A methodology is presented for both obtaining a metric that allows comparison of different modes of operation for any loss of data, and for the choice of parameters of the controllers implemented in each node of the distributed control system. Multi-mode which allows adapting the system developed at different situations arise. The work was completed with the implementation of distributed system simulation and test of a real process.

Keywords:

Embedded systems, model predictive control, distributed control systems.

Agradecimientos

Este trabajo ha sido parcialmente financiado por el proyecto CICYT COBAMI con referencia DPI 2011-28507-C02-01/02, del Ministerio de Educación y Ciencia.

Referencias

Albertos, P., Crespo, A., Simó, J.E., 2006. Control Kernel: A key concept in embedded control systems. 4th IFAC Symposium on Mechatronic Systems.

Albertos, P., Crespo, A., Vallés, M., Ripoll, I., 2005. Embedded Control Systems: Some Issues and Solutions. 16th IFAC World Congress. Prague, Czech Republic.

Baliga, G., Kumar, P.R., 2005. A middleware for control over networks. Proceedings of the IEEE Conference on Decision & Control, including the Symposium on Adaptive Processes.

Beltrán, J. L., Calabuig, L., Munera, E., Simó, J., & Poza-Lujan, J. L., 2014. Configuration Model for Control Kernel Middleware Based Applications. XXXV Jornadas de Automática.

Benchmark CEA. 2010-2011. Grupo Temático de Ingeniería de Control de CEA. Evaluación de Algoritmos de auto-ajuste de controladores PID. http://www.ceautomatica.es/og/ingenieria-de-control/benchmark-2010-11

Berglind, J. B., Gommans, T. M. P., & Heemels, W. P. M. H., 2012. Self-triggered MPC for constrained linear systems and quadratic costs. In *4th IFAC nonlinear model predictive control conference* (pp. 342-348).

Camponogara, E., Jia, D., Krogh, B. H., & Talukdar, S., 2002. Distributed model predictive control. Control Systems, IEEE, 22(1), 44-52.

COBAMI, 2012-2014. Control jerárquico basado en misiones. DPI2011-28507-C02-01/02

Coronel, J.O. 2007. Porting of RTLinux 3.2rc1 to XScale processors. http://rtportal.upv.es/apps/xscale/. Último acceso: febrero 2014

Coronel, J.O., Blanes, F., Simó, J., Nicolau, V., 2008. Middleware de Kernel de Control Para el Desarrollo de Aplicaciones en Sistemas Empotrados de Tiempo Real. XXIX Jornadas de Automática, Tarragona.

Crespo, A., Albertos, P., Balbastre, P., Vallés, M., Lluesma, M., Simó, J.E., 2006. Schedulability issues in complex embedded control systems. Proceedings of the 2006 IEEE Conference on Computer Aided Control Systems Design. Munich, Germany.

Dorf, R., Bishop, R. H., 1995. Modern Control Systems, Seventh Edition. Addison-Wesley.

Eqtami, A., Dimarogonas, D. V., & Kyriakopoulos, K. J., 2011. Event-triggered strategies for decentralized model predictive controllers. In IFAC World Congress (pp. 10068-10073).

Farokhi, F., Shames, I., & Johansson, K. H., 2014. Distributed MPC via dual decomposition and alternative direction method of multipliers. In Distributed Model Predictive Control Made Easy (pp. 115-131). Springer Netherlands.

García Valls, M., Rodríguez López, I., Fernández Villar, L., 2013. iLAND: An Enhanced Middleware for Real-Time Reconfiguration of Service Oriented Distributed Real-Time Systems. IEEE Transactions on Industrial Informatics, Vol. 9, nº 1, (pp 228-236)

Giselsson, P., & Rantzer, A., 2010. Distributed model predictive control with suboptimality and stability guarantees. In Decision and Control (CDC), 2010 49th IEEE Conference on (pp. 7272-7277). IEEE.

Giselsson, P., & Rantzer, A., 2014. On feasibility, stability and performance in distributed model predictive control. Automatic Control, IEEE Transactions on, 59(4), 1031-1036.

Heemels, W. P. M. H., Johansson, K. H., & Tabuada, P., 2012. An introduction to event-triggered and self-triggered control. In CDC (pp. 3270-3285).

KERTROL, 2006-2008. Núcleo de control en los sistemas empotrados fuertemente interconectados. DPI2005-09327-C02-01/02

Kyoung-Dae Kim and P.R. Kumar, 2012. Cyber-Physical Systems: A Perspective at the Centennial. Proceedings of the IEEE. Vol. 100 (pp.1287-1308)

Li, Q., Yao, C. 2003. Real-Time Concepts for Embedded Systems. CMP Books.

Muñoz, M., Munera, E., Blanes, J. F., Simo, J. E., & Benet, G. (2013). Event driven middleware for distributed system control. XXXIV Jornadas de Automatica, 8.

Ohlin, M., Henriksson, D., Cervin, A., 2007. TrueTime 1.5 - Reference Manual. Lund University: Department of Automatic Control.

Poza, J., Posadas, J., & Simó, J., 2009. From the Queue to the Quality of Service Policy: A Middleware Implementation. Distributed Computing, Artificial Intelligence, Bioinformatics, Soft Computing, and Ambient Assisted Living, 432-437.

Rajkumar, R., Lee, I., Sha, L., Stankovic, J., 2010. Cyber-Physical Systems: The Next Computing Revolution. Design Automation Conference (DAC), 47th ACM/IEEE. Anaheim, CA, pp. 731-736.

SIDIRELI, 2009-2011. Sistemas distribuidos con recursos limitados. Núcleo de control y coordinación. DPI2008-06737-C02-01

Simarro, R., 2011. Núcleo de control y diseño de controladores modulares en un entorno distribuido. Tesis doctoral. Departamento de informática de sistemas y computadores (DISCA). Universidad politécnica de Valencia.

Simarro, R., Coronel, J.O., Simó, J.E., Blanes, J.F., 2008. Hierarchical and Distributed Embedded Control Kernel. IFAC 2008 World Congress. Seoul, Korea.

Simarro, R., Albertos, P., Simó, J.E., 2013. Control kernel based adaptive control implementation. SIGBED review. 10, pp. 24-28. ISSN 1551-3688. DOI: 10.1145/2492385.2492389

Yodaiken, V. and Barabanov, M., 1997. Introducing Real-Time Linux. Linux Journal