SYNERGY, PLATAFORMA DE MONITORIZACIÓN BASADA EN CORBA. PRIMERA APROXIMACION AL DISEÑO DE UN CONTROLADOR CON RESTRICCIONES DE COMUNICACION.

Karina Cantillo [1]
Rodolfo E. Haber[1,2], Jose E. Jimenez[1], Ángel Alique[1]
[1]Instituto de Automática Industrial – CSIC, Campo Real Km. 0.200,
Arganda del Rey, Madrid 28500
{cantillo, rhaber, jejc, a.alique}@iai.csic.es
[2] Escuela Politécnica Superior. Ciudad Universitaria de Cantoblanco
Ctra. de Colmenar Viejo, km. 15. 28049 - SPAIN
Rodolfo.Haber@ii.uam.es

Resumen

La combinación de los distintos paradigmas de computación distribuida con el continuo desarrollo de las infraestructuras de comunicación, está cambiando de forma radical los enfoques de diseño de los sistemas de computación, basándose en nuevos modelos de flujos de datos fundamentados en el concepto de recursos computacionales compartidos. Es evidente la influencia positiva de estos avances al área del control, lo cual define una alternativa de desarrollo para la nueva generación de sistemas de control en red, sustentada en tecnologías de computación y comunicación de uso común y bajo coste. El objetivo principal de este trabajo es el desarrollo de SYNERGY, una plataforma de software abierta para la monitorización en red, aplicada a procesos electromecánicos complejos (i.e. Mecanizado de Alta Velocidad), la cual considera los requerimientos del control con restricciones de comunicación. La plataforma está implementada sobre el estándar de computación distribuida tiempo real, *Real Time* CORBA. Los resultados preliminares de la monitorización en red en tiempo real de los procesos de mecanizado a alta velocidad son satisfactorios, a pesar de las restricciones de acceso del CNC (*Computerized Numeric Control*). Dichos resultados corroboran la viabilidad de SYNERGY para la monitorización, supervisión y control en red, con base en las actuales tecnologías de computación y comunicación.

Palabras claves: CORBA, Sistemas con Restricciones de Comunicación, Sistemas de tiempo real.

1 INTRODUCCIÓN

La nueva era de la información basada en las innovadoras infraestructuras de comunicación, ha generado el desarrollo de distintos paradigmas de computación en red y la cimentación de muchos ya existentes, basándose fundamentalmente en el concepto de recursos computacionales compartidos enmarcados en las nuevas tecnologías de comunicación.

Estos paradigmas engloban distintos aspectos de computación y comunicación, definiendo así una unidad conceptual asociada a distintas áreas de la información como Internet, bases de datos, inteligencia artificial, entre otras, lo cual ha contribuido a la participación activa de muchos organismos, tanto públicos como privados, en líneas de investigación sobre la aplicabilidad de dichos paradigmas en sus campos de interés. Muchas de las líneas definidas intentan solventar distintas limitaciones computacionales generadas por los actuales entornos informáticos, para suplir demandas como mayor capacidad de cómputo, la mejora en los servicios Web, el desarrollo de software para entornos distribuidos, el acceso oportuno a la información, etc.

Una de las principales ganancias a obtener con el uso de estos paradigmas es el acceso oportuno a los recursos computacionales desde distintos puntos geográficos, abriendo con esto los campos de aplicación a sistemas en red de tiempo real. Es evidente la influencia positiva de estos avances al área de los sistemas de control, incrementando el uso de infraestructuras de conexión que proveen una mayor disponibilidad de la información para todos los dispositivos de control conectados a un entorno distribuido, conocido con el nombre de Sistemas de Control en Red [2] ó Control Con Restricciones de Comunicación [18], lo cual facilita el diagnóstico, la monitorización y el mantenimiento de los procesos, además de la reducción de costos de cableado, partes y otros.

Es evidente las rigurosas restricciones tiempo real que posee este tipo de sistemas, con requisitos que

afectan enormemente su rendimiento y estabilidad, como son: frecuencia de muestreo, tiempos de respuesta mínimos y capacidad de procesamiento alto, estando sustentadas por tecnologías de conexión costosas para asegurar un comportamiento determinista a nivel de software y hardware [8]. Como ejemplo se tienen los protocolos de control en red: Control Area Network (CAN), Profibus, FieldBus, DeviceNet y otros. Una alternativa acertada y baja en costos, de las tecnologías de conexión antes mencionadas, la está constituyendo las redes de datos de uso común como *Ethernet*, por la continua evolución que sobre éstas se está dando, tanto en velocidad como en ancho de banda.

Una posible solución a los aspectos tiempo real demandados la provee el desarrollo de estándares basados en los paradigmas de computación distribuidas tales como GRID *Computing* [3], CORBA [1], etc., los cuales convierten los aspectos de calidad de servicio, como: demoras, variación de demoras, fiabilidad y rendimiento predecible, en propiedades controlables a nivel de usuario. Esto en combinación con las distintas técnicas de identificación y reacción de procesos, y la posible accesibilidad a los recursos de computación distribuidos, forman la base para el desarrollo de la nueva generación de sistemas de control distribuidos, sustentados en tecnologías de computación y comunicación de uso común (i.e., los sistemas operativos Windows, el protocolo TCP/IP, etc.), reduciendo con esto el coste de implementación, aumentando la flexibilidad del montaje, de actualización y la aplicabilidad de las tecnologías de la computación y comunicación en surgimiento.

El objetivo principal de este artículo es presentar los resultados relacionados con el diseño y la implementación de una plataforma de software abierta llamada SYNERGY, para la monitorización distribuida, como una primera aproximación al desarrollo de controladores con restricciones de comunicación, por medio de la cual se obtendrá información relevante para el análisis de factibilidad relacionado con los sistemas de control en red. La plataforma fue desarrollada en base al estándar tiempo real CORBA, y es aplicado a procesos electromecánicos complejos (i.e. Mecanizado de Alta Velocidad). Una de las principales contribuciones de este trabajo, es el desarrollo de una plataforma de software basada en tecnologías de uso global, con bajo coste de implementación y mantenimiento, considerando al mismo tiempo los requerimientos respecto a la monitorización, supervisión y control en red.

Este trabajo se organiza como se describe a continuación. En la segunda sección se enuncian los aspectos claves relacionados con las principales restricciones de los sistemas de control en red. En la tercera sección se argumenta sobre la elección del estándar RT-CORBA como herramienta de implementación de la plataforma distribuida. En la cuarta sección se realiza una breve descripción de los procesos de mecanizado a alta velocidad y la descripción de los aspectos de la arquitectura desarrollada para la medición en tiempo real de las variables características del proceso. La quinta sección está relacionada con el diseño y la implementación de la plataforma de software SYNERGY y los resultados preliminares concernientes a la monitorización de los procesos de mecanizado a alta velocidad. Al final se enuncian las principales conclusiones del trabajo realizado.

2 SISTEMAS CON RESTRICCIONES DE COMUNICACIÓN

El progreso creciente en las infraestructuras de comunicación y su amplia utilización en todas las áreas del conocimiento existentes, ha propiciado su uso en las áreas de control, estableciéndose así una alternativa de desarrollo de controladores en red, extendidos geográficamente en grandes distancias, sin requerir el soporte de costosas infraestructuras de comunicación.

Sin importar que protocolo e infraestructura de red se utilice, el rendimiento de un sistema de control con restricciones de comunicación siempre estará afectado por el comportamiento de las demoras de red. Varios han sido los enfoques de conexión utilizados para el diseño de controladores con restricciones de comunicación, orientados a proveer las características propias del tráfico de control: envío secuencial y en pequeños paquetes (cuantificación), teniendo en cuenta los requerimientos tiempo real del proceso. Se definen dos topologías de conexión: control directo y control jerárquico [17]. En la topología directa, la salida del proceso es directamente conectada al controlador remoto y la salida de dicho controlador es enlazada al actuador del sistema, todo a través de la red de datos. En la estructura jerárquica existe un controlador local enlazado al Sensor–Proceso-Actuador, y un controlador principal conectado en red, que se encarga de establecer ciertos parámetros del controlador local.

Existen distintas metodologías de diseño de controladores en red, sustentadas todas en los requerimientos tiempo real del proceso y el comportamiento del tráfico de red. Como ejemplo se tiene un diseño basado en un pronosticador remoto, que toma la señal que viene del sensor y teniendo en cuenta la demora de red, predice el estado actual y el estado futuro del sistema, la cual es pasada al

controlador que elige la respectiva acción, ésta es enviada al actuador. Para este ejemplo se hace necesario un modelo preciso del proceso, sobre el cual se fundamenten tanto el pronosticador como el controlador. Este diseño se basa en técnicas de estabilidad, que compensan la degradación del sistema por las demoras de comunicación existentes entre Sensor-Controlador-Actuador. Como esta metodología de compensación de demoras existen muchas otras, tales como las metodologías de control robusto, control borroso, control de adaptación del usuario final, etc. En esta última, tanto el controlador como el sensor miden las demoras del tráfico de red, realizando con éstas la adaptación de las ganancias, obteniendo así un mejor rendimiento.

Es primordial, en el camino hacia el diseño óptimo de un control en red, garantizar la transmisión de la información y establecer puntualmente políticas de reducción y limitación de las demoras de comunicación. Desde esta perspectiva se aprecia la importancia de una completa exposición de los principales aspectos concernientes al comportamiento del tráfico de red y las demoras de comunicación en los entornos de computación existentes. Por medio de un modelo de las demoras de red de la plataforma distribuida propuesta, se podrá establecer el grado de factibilidad y fiabilidad del controlador remoto a diseñar. Una alternativa de implementación estaría determinada por la demora obtenida en el modelo, que en combinación con el modelo del proceso, mostraría el grado de afectación del comportamiento de la red a los requerimientos tiempo real del sistema. Adicionalmente, se podrían definir políticas de optimización de la tasa de muestreo que haga cumplir con las características del flujo de los sistemas de control, sin ser causa de congestión de red.

Existen muchos tipos de modelos utilizados para el tráfico de red, están los analíticos, empíricos, probabilísticos, teóricos, etc. Entre los modelos analíticos existentes está el modelo de un paquete y el modelo de dos paquetes [6]. De éstos se tiene que a pesar de realizar un buen análisis del comportamiento del tráfico de red, pudiendo establecer causas y consecuencias, no logra representar el comportamiento del tráfico de red en propiedad, debido principalmente a que las hipótesis sobre las cuales se sustenta, no se ajustan a la realidad, dando como resultados simulaciones erróneas. Por otro lado se tienen los modelos probabilísticos, como los modelos de Poisson, las cadenas de Markov, los modelos de series temporales, los modelos de cola pesada, los modelos auto-similares [7], entre otros, que a pesar de ser herramientas tipo caja negra con las cuales no se pueden establecer causas ni consecuencias de las conductas características de la red, reproducen con un alto nivel de fiabilidad el comportamiento del tráfico. Otros aspectos a tener en cuenta son el grado de generalidad y de simplicidad del modelo, para que su alcance representativo abarque una gran variedad de servicios de red (audio, video, datos, multimedia, etc.) y a su vez requiera el menor número de parámetros de descripción.

Con un modelo de demoras ajustado a las características del comportamiento del tráfico de red, se podrán realizar estudios *off-line* de la calidad de la señal obtenida por el cliente remoto, los cuales determinarán las distintas características del controlador a implementar, además de establecer normas de corrección a la señal para que se asemeje a la realidad local en un instante de tiempo determinado, que compensen el desfase temporal variante de la transmisión de señal por una tecnología de comunicación de comportamiento estocástico como las redes *Ethernet*. Éste sería el resultado analítico de la combinación entre el modelo del proceso y el modelo de demoras. Todo esto ha de ser clave al diseño e implementación de la plataforma distribuida a desarrollar.

3 IMPLEMENTACIÓN DE SISTEMAS DISTRIBUIDOS DE TIEMPO REAL. *REAL TIME* CORBA

A pesar que los sistemas de control a gran escala cuentan con sistemas de computación y comunicación de comportamiento predecible (i.e. Sistemas de Manufactura Industrial), se hace necesario el mejoramiento progresivo de los sistemas distribuidos orientados a objetos, enfocándose en la gestión de las restricciones de comunicaciones. Para suplir estas necesidades en los distintos dominios de aplicación, como el aeroespacial, automotriz, etc., se ha incrementado el uso de *Middleware* en la computación de objetos distribuidos. El *Middleware* CORBA (*Common Object Request Broker Architecture*), especificación definida por la OMG (*Object Management Group*), estructurada en capas, núcleo ORB (*Object Request Broker*), servicios CORBA y capa de aplicación, provee abstracciones y servicios que proporcionan una capa homogénea de desarrollo de aplicaciones distribuidas, eximiendo a los programadores de complejas tareas de gestión a bajo nivel para las diversas plataformas existentes en los entornos de computación.

Con el objetivo de soportar los requerimientos tiempo real de los distintos campos de aplicación, se define la especificación *Real Time* CORBA, la cual define mecanismos y políticas que permiten controlar los recursos del procesador, los recursos de comunicación y los recursos de memoria.

Una de las partes de la arquitectura CORBA responsable de un mejor rendimiento de las aplicaciones CORBA es el núcleo del ORB. Éste establece las conexiones por parte del cliente y atiende las solicitudes por parte del servidor. La implementación de estas tareas puede convertir al núcleo del ORB en fuente activa de indeterminismo, lo cual ha servido de referencia para la elección de la implementación CORBA a utilizar. El ORB tiempo real debe evitar el establecimiento de conexiones dinámicas, reducir la gestión de memoria dinámica y evitar la multiplexación de solicitudes de diferente prioridad en una conexión compartida. El ORB de TAO (The ACE ORB), a diferencia de la mayoría de las implementaciones CORBA del mercado (MT-Orbix, CORBAplus, Visiobroker, miniCOOL, OrbixWeb, Java IDL, TAO, MICO, Ica, OmniORB, eORB), ofrece estas características por omisión. El núcleo del ORB de TAO comparte una mínima cantidad de recursos del ORB, lo cual reduce substancialmente el coste de sincronización y la inversión de prioridad entre los hilos de un proceso [14]. TAO es una implementación tiempo real CORBA, de libre distribución, con código fuente abierto, construido en el marco de componentes y patrones provistos por ACE (The ADAPTIVE Communication Environment) [13].

El diseño y la implementación de una plataforma software para la monitorización, supervisión y control en red de sistemas dinámicos (i.e. Procesos de Mecanizado a Alta Velocidad), requiere la consideración de importantes tópicos relacionados con la teoría de control y de comunicación (i.e. Teorema de Shannon). Como se ha mencionado en la sección anterior, los controladores robustos están diseñados con base en las demoras fijas concernientes a los procesos de medición y actuación. Sin embargo en los sistemas de tiempo real distribuidos, el comportamiento estocástico de las demoras computacionales y de comunicación, conduce a la degradación del sistema. Estos aspectos de diseño fueron considerados para la elección de ACE-TAO como *Middleware* durante el desarrollo de la plataforma.

Por otro lado, la diversidad de los campos de aplicación para los sistemas de tiempo real define una clasificación en función de los requerimientos. Los sistemas estrictos de tiempo real, en el cual las demoras no definidas son inaceptables y pueden generar fallos catastróficos en el sistema, requieren el soporte de una plataforma tiempo real estricta. En los sistemas de tiempo real flexibles, en el cual son aceptadas algunas demoras no definidas compensadas con la degradación del sistema, es suficiente el uso de sistemas operativos de propósito general.

El resultado de algunas investigaciones sobre la medición de los parámetros de calidad de servicio como latencia, *jitter*, costos de procesamiento de la CPU, etc., con sencillas aplicaciones cliente – servidor sobre las plataformas de mayor popularidad (i.e. Windows NT, LynxOS, Linux, Solaris, VxWorks), muestran que los sistemas operativos de propósito general generan altas latencias e indeterminismo en su comportamiento, en comparación con los sistemas operativos tiempo real [12].

Por ejemplo, el comportamiento no-determinista de los sistemas operativos Windows 2X se debe al bajo número de prioridades a asignar a los hilos de un proceso, el inadecuado uso de la memoria *cache*, la falta de soporte al protocolo *priority inheritance* en los mecanismos de sincronización, la ejecución de las tareas de entrada-salida sin importar el nivel de prioridad del hilo que las requiere, etc. A pesar de esto, Windows 2X es un sistema operativo multitarea con rápidas respuestas que llena los requisitos de un sistema de tiempo real flexible. Posee una API Win32 muy robusta para los programadores, provista de muchas herramientas de desarrollo, buenos compiladores, una interfaz de usuario gráfica de fácil uso y una buena administración de memoria principal. Adicionalmente, Windows 2X es una de las plataformas más utilizadas a nivel de usuarios, en empresas y hogares. Con estas ventajas y otras características del sistema operativo, se hace viable su utilización como soporte para el desarrollo de aplicaciones tiempo real flexible.

4 MECANIZADO A ALTA VELOCIDAD

El mecanizado a alta velocidad, MAV, es una de las tecnologías claves para el desarrollo de importantes sectores productivos tales como la automoción, moldes y matrices, la aeronáutica, máquina herramienta, etc., representando un buen porcentaje de la economía de los países que disfrutan del conocimiento de ésta. Uno de los objetivos más perseguidos por la comunidad científica en el mecanizado a alta velocidad es la obtención de patrones del esfuerzo de corte. Esto se debe a su aporte para el establecimiento de criterios de deflexión, desgaste y rotura de la herramienta de corte. Adicionalmente también ofrece información acerca de la calidad y el perfil geométrico de la superficie de corte [14]. La respuesta oscilatoria y picos en los patrones de esfuerzo de corte pueden ser el resultado de sobrecargas, lo que indicaría la irregularidad del proceso y el incremento de peligro de rotura de la herramienta o daños en la pieza de trabajos. De acuerdo a los análisis realizados del proceso de desgaste, las variables de interés de los procesos de mecanizado a alta velocidad son: La

posición espacial de la herramienta de corte, considerando los ejes cartesianos (x_p, y_p, z_p) [*mm*], la velocidad del cabezal *(s)*[*rpm*], la velocidad de avance entre la herramienta y la mesa de trabajo (*f*)[*mm/min*], la potencia de corte del cabezal investida en la eliminación de viruta de metal de la pieza de trabajo (P_c)[*kW*], el esfuerzo de corte aplicado durante la extracción de la viruta de metal (*F*)[*N*], la profundidad radial de corte (*a*)[*mm*], el diámetro de la herramienta de corte (*d*)[*mm*].

Se ha escogido el proceso de fresado como caso de estudio, siendo éste la operación de mecanizado más compleja debido a las características no lineales intrínsecas del proceso. Las pruebas han sido realizadas en el centro de mecanizado de alta velocidad marca KONDIA HS-1000 con CNC Siemens SINUMERIK 840D. La conexión entre los componentes funcionales del CNC (*Numerical Control Unit*-NCU, *Man Machine Communication*-MMC) y la plataforma SYNERGY se realiza por medio de la interfaz MPI (*Multiport Interface*). Se han instalado sensores externos (seis acelerómetros y dos sensores de emisión acústica) en puntos clave del centro de mecanizado. Las señales de los sensores son adquiridas y procesadas utilizando un programa basado en *Labview* llamado *SignalAcquisition*. En la figura 1 se muestra la arquitectura de adquisición antes descrita.

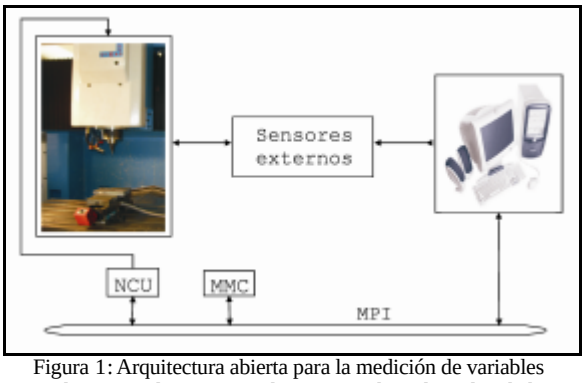

Figura 1: Arquitectura abierta para la medición de variables relevantes a los procesos de mecanizado a alta velocidad

Actualmente, la alternativa más accesible para la adquisición de datos del centro de mecanizado en tiempo real, es el establecimiento de una conexión *DDE* (*Dynamic Data Exchange*) con la aplicación *NCDDE Server*, provista por Siemens. La aplicación *NCDDE Server* implementa tres servicios de acceso al control numérico [15]: servicio de variable simple, servicio de variable vector, servicio de multi-variable. Con el objetivo de obtener la configuración de acceso más eficiente entre los distintos servicios disponibles por la aplicación *NCDDE Server*, se implementó un programa en Microsoft Visual C++ 6.0 para la obtención de las distintas demoras de acceso. Los archivos resultantes muestran una reducción de aproximadamente 60% en la demora de adquisición de todas las variables de interés con el servicio multi-variable del *NCDDE* Server.

5 RESULTADOS

Fue desarrollada una plataforma para la monitorización, supervisión y control en red del mecanizado a alta velocidad, basada en objetos distribuidos. Esta plataforma está diseñada para la incorporación de una arquitectura de software que habilite la conexión entre el centro de mecanizado + CNC con la aplicación remota. El diseño e implementación de la plataforma de tiempo real contribuye a la integración en corto plazo, de metodologías clásicas y de inteligencia artificial en la aplicación remota.

La plataforma de software SYNERGY consta de dos aplicaciones, una aplicación servidor llama *Monitoring-Server* y una aplicación cliente conocida como *Remote-Monitoring*. Cada aplicación está constituida por varios módulos. La aplicación *Monitoring-Server* posee el modulo de adquisición de datos, el módulo de identificación y el módulo de comunicaciones. La aplicación *Remote-Monitoring* costa del módulo de comunicación y el módulo de control. Este trabajo está enfocado en los módulos de adquisición de datos y de comunicaciones, tanto de la aplicación servidor como la cliente. La figura 2 muestra la plataforma diseñada.

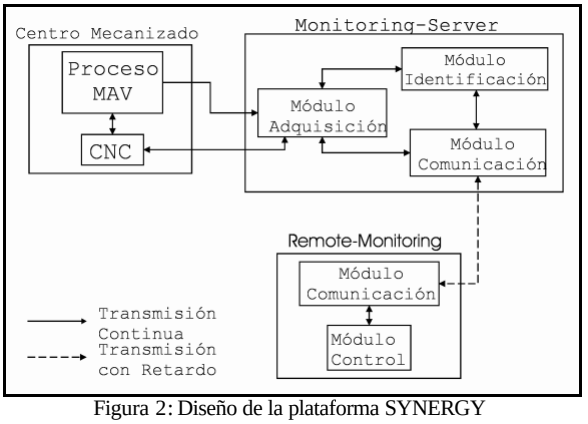

Figura 2: Diseño de la plataforma SYNERGY

El módulo de comunicación de ambas aplicaciones se desarrolló en base al estándar CORBA, utilizando la implementación TAO. Se hizo uso inicialmente de la versión ACE5.2 + TAO 1.2 y luego por fallos propios de la versión, se migró a las versiones ACE5.3 + TAO1.3. El entorno de programación escogido es Microsoft Visual C++ 6.0 profesional (MSVC++ 6.0), debido a su amplia utilización en aplicaciones industriales, el amplio soporte a nivel de ayudas, componentes especializados y la gran cantidad de desarrollos existentes en este lenguaje de programación. La siguiente es la interfaz de

comunicación definida para las aplicaciones CORBA.

```
module rtcontrol{
typedef sequence<string> vectdatos;
interface monitor{
  string solicitud(in short item);
  long inidatosloop();
  void datosloop(in long indini,out
long indend, in short item, out
vectdatos vcdts);
};

interface controller{
  typedef sequence<vectdatos,6>
muestra;
  void iniobtmuestra(out long ind);
  long obtmuestra(in long ind,out
muestra mact);
};
};
```

Entre los servicios y las funciones TAO implementados en las aplicaciones están: el servicio de nombres, para la localización de objetos CORBA, y los siguientes mecanismos y políticas *Real Time* CORBA: *Real-Time* ORB, *Real-Time* POA, *Real-Time* Current, *Priority Mappings*, *Server_Declared Priority Model*, *Server Protocol Policy*, *Explicit Binding*, *Private Connections*, POA *Threadpools*. A continuación se muestra una parte del código fuente de la aplicación servidor desarrollada.

```
// RTORB.
CORBA::Object_var obj = orb-
>resolve_initial_references("RTORB");
RTCORBA::RTORB_var rt_orb =
RTCORBA::RTORB::_narrow(obj.in());
RTCORBA::ThreadpoolId conthreadpool_id=
rt_orb-
>create_threadpool(0,4,0,30000,0,0,0);

//crear lista de políticas
.......
rt_orb->
create_priority_model_policy(RTCORBA::SERVER
_DECLARED,30000/*HIGH_PRIORITY*/);
```

El módulo de adquisición de la aplicación *Monitoring-Server* se comunica por *DDE* con las aplicaciones *NCDDE Server* y *SignalAcquisition*. Los valores de las variables son almacenados en una matriz cíclica para no saturar el espacio en memoria. La principal tarea de los métodos implementados de los objetos CORBA es acceder a la matriz cíclica, obtener los últimos valores almacenados y enviarlos al cliente que realizó la petición. En la siguiente figura describe lo anterior.

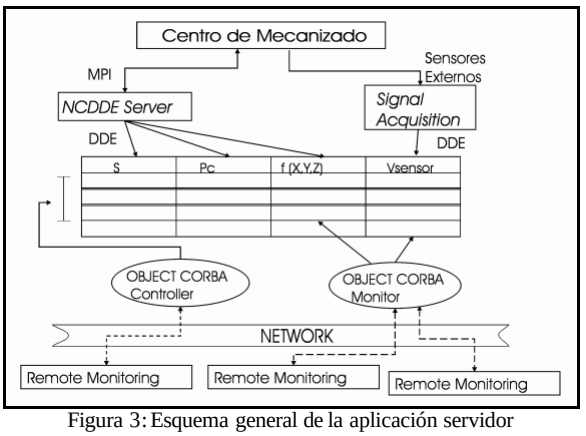

Figura 3: Esquema general de la aplicación servidor

La plataforma desarrollada fue evaluada con un mecanizado a alta velocidad en tiempo real. Para tales pruebas se utilizaron tres herramientas con distintas características de desgastes: Herramienta nueva, herramienta semi-desgastada, herramienta desgastada. El programa de mecanizado consistió de nueve ranuras en una pieza de aluminio, con una profundidad inicial de 0.5 mm.. y un incremento de 0.5 mm en cada pasada.

Los experimentos fueron realizados considerando los índices de tráfico de red, para evaluar el efecto del comportamiento de ésta en el rendimiento de la aplicación. Tanto la aplicación cliente como la aplicación servidor, almacenan archivos de datos referente a la matriz cíclica. Los archivos de la aplicación cliente poseen una columna adicional donde se registran las demoras de comunicación de cada petición de datos. Por cuestión de espacio solo dos casos son mostrados en la figura 4.

La figura 4a y 4c muestra el comportamiento de la velocidad del cabezal (línea a trozos), el avance en el eje X (línea de puntos), el avance en el eje Z (línea en negrita) y el avance en el eje Y (línea a trozos y puntos), para los índices de tráfico de red alto y bajo. La figura 4b y 4d muestra el comportamiento de las demoras de comunicación con índices de tráfico de red alto y bajo respectivamente. Uno de los ensayos realizados con alto flujo en la red obtuvo una demora media de 0.01089 seg. con una varianza de 0.0006897 seg. En presencia de bajo flujo de tráfico en la red se obtuvo una media de 0.008826 seg. y una varianza de 0.0002312 seg.

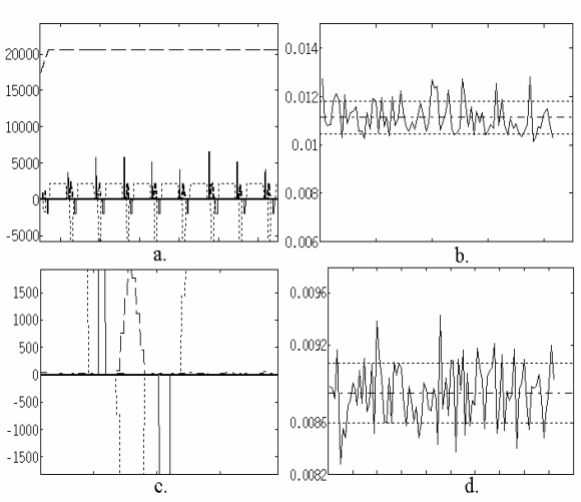

Figura 4: Comportamiento de las principales variables del proceso y las demoras de comunicación en presencia de altos y bajos índices de tráfico de red

6. CONCLUSIONES.

Ha sido desarrollada una arquitectura abierta para la medición de las principales variables de los procesos de mecanizado a alta velocidad. Adicionalmente ha sido diseñada la plataforma de software SYNERGY para la monitorización en red, como una primera aproximación al diseño del controlador en red de los procesos de mecanizado a alta velocidad.

Después del estado del arte realizado en las distintas áreas del conocimiento involucradas en este trabajo, y teniendo en cuenta el análisis de los resultados de las pruebas realizadas, se concluye preliminarmente:

- Es viable el desarrollo de una plataforma de pruebas basada en las actuales tecnologías de comunicación y computación como los sistemas operativos de la familia Windows 2X, la arquitectura CORBA, el protocolo TCP/IP, etc. Esto mantiene la portabilidad del sistema obtenido, el bajo coste de diseño e implementación, su utilización en múltiples campos de aplicación y la facilidad de adicionar nuevas tecnologías desarrolladas en las áreas de investigación involucradas.
- El tráfico de control posee una conducta periódica, definido a valores mínimos, con paquetes de corto tamaño. Estas características no se ajustan al comportamiento de ciertas tecnologías de redes como las *Ethernet*, ocasionando saturación en los canales de transmisión, llevando a la degradación del comportamiento de los sistemas de control en red. Para estos caso, se deben definir ciertas políticas en las aplicaciones clientes, que permitan adaptar la transmisión de datos a las características requeridas por dichos sistemas. Por medio del modelo de las demoras de comunicación se podría establecer un tiempo de muestreo óptimo, que aprovechara el ancho de banda existente sin ocasionar sobrecarga en la red.
- El ORB Real Time TAO, a diferencia de la mayoría de las implementaciones CORBA del mercado, muestra supremacía en el desarrollo de aplicaciones CORBA tiempo real, debido a su diseño en las arquitecturas de conexión y de concurrencia, pudiéndose obtener un comportamiento determinista y predecible en el envío y atención de solicitudes.
- Es necesario aumentar la frecuencia de muestreo por parte del módulo de adquisición de datos implementado, lo cual es posible con la continua expansión de las tecnologías CNC y los dispositivos relacionados.

Referencias

[1] Beckwith, B.: Real Time CORBA Tutorial. Objective Interface Systems, Inc. OMG Real Time and Embedded Workshop (2002)

[2] Branicky, M.S., Phillips, S.M., Zhang, W.: Stability of Networked Control System: Explicit Analysis of Delay. Proceedings of the American Control Conference (2000) 2352-2357

[3] Greenberg, J.P., Mock, S., Katz, M., Bruno, G., Sacerdoti, F., Papadopoulus, P., Baldridge, K.K.: Incorporation of Middleware and Grid Technologies to Enhance Usability in Computational Chemistry Applications. 4th International Conference of Computational Science. Vol. 1 (2004) pp. 75-82

[4] Haber, R.E., Peres, C.R., Alique, A., Ros, S., Alique, J.R.: Towards intelligent machining: hierarchical fuzzy control for end milling process IEEE Trans. Control Syst.Technol. Vol. 2, 6 (1998) pp 188-199

[5] Hristu, D., Morgansen, K.: Limited Communication Control. Systems & Control Letters, Elsevier Science B.Vol. 37 (1999) 193-205

[6] Lai, K., Baker, M.: Measuring link bandwidths using a deterministic model of packet delay. Proceedings of *ACM SIGCOMM*, Stockholm, Sweden. (2000) pp. 283 – 294

[7] Leland, H. E., Taqqu, M. S., Willinger, W., Wilson, D. V.: On the self-similar nature of Ethernet traffic. IEEE/ACM Trans. Networking, Vol. 2 (1994) pp. 1-15

[8] Lian, F-L., Moyne, J. R., Tilbury, D. M.: Control Performance Study of a Networked Machining Cell. Proceedings of 2000 American Control Conference, Chicago, IL. Vol. 4 (2000) pp. 2337-2341

[9] Low, S.H., Paganini, F., Doyle, J.: Internet Congestion Control. IEEE Control Systems Magazine (2002)

[10] Park, H.S., Kim, Y.H., Kim, D.-S., Kwon, W.H.: A Scheduling Method for Network-Based Control Systems. IEEE Transactions on Control Systems Technology, Vol. 10, 3 (2002)

[11] Pyarali, I., Schmidt, D.C., Cytron, R.: Achieving End-to-End Predictability of the TAO Real-time CORBA ORB. Proceedings of the 8th IEEE Real-Time Technology and Applications Symposium, San Jose, CA (2002)

[12] Schmidt, D.C., Deshpande, M., O'Ryan, C.: Operating System Performance in Support of Real-time Middleware. Proceedings of the 7th IEEE Workshop on Object-oriented Real-time Dependable Systems, San Diego, CA (2002)

[13] Schmidt, D.C., Levine, D.L., Mungee, S.: The Design of the TAO Real Time Object Request Broker. Computer Communications, Elsivier Science, Vol. 21, 4 (1998)

[14] Schmidt, D.C., Mungee, S., Gaitan, S.F, Gokhale, A.: Software Architectures for Reducing Priority Inversion and Non-determinism in Real-time Object Request Brokers. Journal of Real-time Systems, special issue on Real-time Computing in the Age of the Web and the Internet (2001)

[15] SINUMERIK 840D/810D/FM-NC. User Manual. Edition 09 2000, Siemens

[16] TAO Developer's Guide, Building a Standard in Performance. Object Computing, Inc. TAO version 1.2 a, Vol. 1, 2. St. Louis (2002)

[17] Tipsuwan, Y., Chow, M-Y.: Control Methologies in Networked Control Systems. Department of Electrical and Computer Engineering, North Carolina State University, USA. Control Engineering Practice, Elsevier, 11 (2003) 1099 – 1111

[18] Wong, W.S., Brockettt, R.W.: Systems with Finite Communication Bandwidth Constraints II. Stabilization with Limited Information Feedback. IEEE Transactions on Automatic Control, Vol. 44, 5 (1999) 1049-1053