# UN MODELO DE EVALUACION DEL DESEMPEÑO DE SISTEMAS CLIENTE-SERVIDOR*

Francisco Aurtenechea; Gabriel Rosenberg; Ignacio Casas

Pontificia Universidad Católica de Chile
Depto. de Ciencia de la Computación
Casilla 306, Santiago, Chile
FAX: (5-62) 5524054 email: fa@puc.cl

## RESUMEN

En este trabajo se presentan metodologías y herramientas para la evaluación y predicción del desempeño de aplicaciones cliente/servidor a través de redes locales Ethernet. El objetivo es evaluar el impacto de los distintos niveles de desempeño de la red en el desempeño global de este ambiente transaccional. Se presenta el desarrollo de herramientas de comunicación como medio ambiente de monitoreo y medición que, junto a modelos analíticos de desempeño, permitan proyectar el impacto de los diferentes niveles de servicio de la red en aplicaciones cliente-servidor. Parte de esta metodología es evaluada mediante mediciones y evaluación del desempeño en el acceso de clientes a una aplicación de base de datos remota.

**Palabras Claves:** Sistemas Cliente-Servidor, Modelación Analítica, Redes Locales.

## I. INTRODUCCION

El auge de las redes locales, se ha traducido en un amplio desarrollo de herramientas de comunicación tendientes a proveer un medio ambiente de operación a estaciones de trabajo. Surgen los sistemas operativos distribuídos [6,15] y esquemas de comunicación cliente/servidor [2,3,15], los que permiten hacer un uso más efectivo de los recursos de la red.

La mayor cobertura de acceso a los recursos distribuídos, causa, además, que la demanda por los recursos de la red, tanto de hardware como de software tengan un aumento sostenido. El monitoreo, evaluación y predicción del desempeño es de vital interés en la evolución de este ambiente computacional. En la medida que los sistemas distribuídos evolucionen y la demanda de uso aumente, el estudio de su desempeño es importante como elemento de diseño y planificación de capacidad.

Hasta la fecha se han desarrollado diversos modelos tanto analíticos como de simulación, para predecir el comportamiento de una red, en función del tráfico o demanda de acceso en la misma [4,5,7,8]. Dichos modelos, en general, analizan aspectos de bajo nivel en la red, sin presentar esquemas comparativos del impacto del tráfico en las distintas interfaces de la red.

En este trabajo se plantea el problema de desagregar la evaluación del impacto de los distintos niveles de interacción de la red en el desempeño de sistemas cliente/servidor. Se proponen metodologías y herramientas de comunicación para el monitoreo y predicción del desempeño de sistemas de acceso distribuído en red local.

Se describe la construcción de un modelo de comunicación, basado en un servidor de aplicaciones [2], tanto para canalizar el acceso, a clientes, a aplicaciones de la red, como para medir el desempeño en las interfaces de comunicación. Mediante la activación controlada de tráfico, se dispone de un monitor de red activo para conocer los retardos en distintos niveles de la red en función del patrón de demanda de aplicaciones específicas.

* Este trabajo fue financiado por FONDECYT, proyecto 0771/91

Se plantea la construcción de modelos de redes de espera [9] para predecir el desempeño de las aplicaciones en distintos escenarios tanto de configuración como de tráfico. Finalmente, se presenta un análisis preliminar de desempeño, mediante la realización de diversas mediciones de tiempos en el acceso a una aplicación remota, en función del número de clientes.

## II. ELEMENTOS DE DESEMPEÑO

Cuando dos nodos se comunican mediante una red, se produce el flujo de transacciones, siendo cada una un mensaje de consulta (desde un cliente) seguido de un mensaje de respuesta (desde un servidor). El valor medio del tiempo de respuesta depende de diversos factores tales como configuración, tráfico y estrategias de diseño. Entre estos factores se incluyen: los medios de comunicación, la topología de la red, los protocolos de comunicación, la capacidad de buffers, velocidades de los procesadores, etc.

El desempeño global de las aplicaciones en una red depende de estos factores y de su mutua interacción. La identificación adecuada de estos factores constituye un requisito indispensable si se desea establecer el desempeño integral de las aplicaciones de la red, junto con la obtención de los factores críticos que afectan su comportamiento

En una red local, los protocolos de comunicación corresponden a una estructura de múltiples niveles que permiten el intercambio de datos entre sus nodos. Cada nivel soporta funciones específicas y a través de interfaces se enlaza con niveles superiores hasta alcanzar las aplicaciones de nivel de usuario. Con respecto a este nivel de comunicación, es necesario conocer:

- El efecto en el desempeño de la red debido a las estrategias de almacenamiento y protocolos de transferencia a través de los diversos niveles funcionales que soportan el diálogo entre clientes y servidores.
- La significación de que cierto procesamiento y actividades de comunicación puedan ser realizadas por hardware o procesadores dedicados (p.ej. en el control de protocolos de nivel intermedio como TCP/IP).
- La importancia de dedicar mayor número de procesos de comunicación a las actividades de transferencia de datos, tanto a nivel de cliente como servidor.

Por otro lado, el uso de un gran ancho de banda en la transmisión de bits seriales es uno de los atractivos de las redes locales. A las técnicas en uso actualmente (p.ej. par trenzado, cable coaxial), se están incorporando otras con un mayor ancho de banda disponible, tal como CATV y fibra óptica. Por lo tanto es necesario establecer lo siguiente:

- Cuan significativo es el ancho de banda en el retardo total de una transacción en una comunicación inter-procesos a través de la red.
- Cuan sensible es el tiempo de acceso a la red al tamaño de la misma, al tamaño de los paquetes, al tráfico. Cuan importante es este tiempo en el desempeño global de las aplicaciones de red.
- El efecto de ciertas estrategias de control de flujo en la red (p.ej. el uso de *acknowledgements* ) en el retardo total de cada transacción.

Finalmente, es necesario considerar factores que son independientes de la red, tal como el tipo de aplicaciones usadas a través de la red, la carga en los servidores, la configuración de los "hosts" (cantidad de memoria, velocidad de dispositivos, etc). La eficiencia de los protocolos de bajo nivel no garantiza un tiempo de respuesta adecuado. Una solicitud a un servidor compartido puede esperar a que otros procesos sean atendidos, o la disponibilidad de controladores de disco, etc. Este retardo puede ser significativo si la carga en la red aumenta. Entonces, es relevante establecer:

- La diferencia de velocidad entre los datos entregados al "host" y la velocidad a la cual éste puede absorberlos.

- Hasta donde puede ser mejorado el desempeño de las aplicaciones, al modificar ciertas estrategias de transferencia de datos (p.ej. uso de buffers dedicados) tanto a nivel de servidor como de cliente.
- Para una aplicación típica, cuan significativo es el tiempo de procesamiento de los datos en la red y en los protocolos, comparado con las actividades de lectura/escritura de datos en el servidor.

Estos y otros factores se deben considerar para proveer una perspectiva más global al evaluar el desempeño de sistemas de acceso distribuído. Es difícil, sin embargo, discernir acerca de todos los factores que influyen en este desempeño, debido a la variabilidad de configuraciones y patrones de tráfico que pueden existir. En la medida que sea posible aislar la mayor cantidad de estos factores, mayor será la posibilidad de conocer y predecir el desempeño de estos sistemas en su dinámica de evolución permanente.

## III. INFRAESTRUCTURA DE EVALUACION

Un usuario conectado a un computador de la red es un cliente remoto para ese computador, el cual destina un proceso (servidor) para atender los requerimientos del cliente. Con la incorporación de sistemas operativos de red, sistemas cliente/servidor, etc, se pone a disposición de los usuarios los recursos de la red en forma casi transparente. La pérdida de transparencia estaría asociada a la contención de la red.

Para medir el desempeño de las transacciones de red en este ambiente de interacción, se desarrolló un sistema servidor de aplicaciones [2]. Esta es una herramienta transparente de acceso a los recursos de la red, e incorpora mecanismos de medición del desempeño en distintos niveles de interacción (interfaces de comunicación) entre un cliente y una aplicación remota.

Estos módulos de comunicación están disponibles tanto en equipos PC-DOS como en equipos Unix de la red (Ultrix y System V). Se utiliza como plataforma de comunicación, una red Ethernet [11], el protocolo TCP/IP y "socket library", permitiendo conectar procesos remotos entre un cliente, asociado a un usuario remoto, y un servidor, asociado a una aplicación remota.

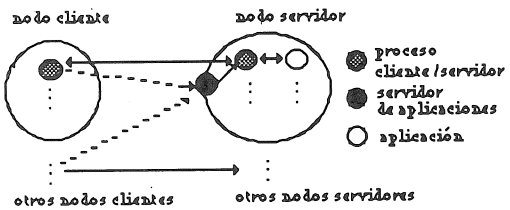


Figura 1. Servidor de aplicaciones

Cuando el servidor de aplicaciones recibe una solicitud de conexión, crea un subproceso y queda en espera de otro cliente, permitiendo a varios clientes interactuar con una aplicación de cualquier nodo de la red. Cada subproceso recibe el nombre del servicio requerido y lo ejecuta, comunicando el "input-output" de dicho servicio con el proceso cliente en el nodo remoto (ver fig.1).

Cada subproceso servidor crea un canal de comunicación para dar acceso, a clientes, a aplicaciones remotas, y re-direcciona la salida de ésta en un "buffer" para ser enviada al cliente remoto (ver fig. 2).Así, cualquier aplicación remota se ejecuta en forma transparente, y manteniendo el *overhead* al mínimo posible en el subproceso servidor.

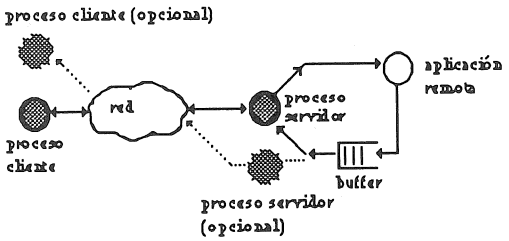


Figura 2.Conexión Cliente/Servidor

## IV. METODOLOGIA DE EVALUACION

La realización de evaluaciones sobre un cierto número de sistemas que interactúan, requiere conocer cada nivel de desempeño. El grado de especificación de cada nivel debe estar en relación a la significación que tiene en el desempeño de los sistemas en niveles sucesivos de interacción.

Si el sub-sistema a evaluar es complejo, entonces se puede reproducir sus niveles de desempeño mediante especificaciones funcionales (i.e. modelos analíticos) de las operaciones más relevantes. El uso de modelos simples para representar sistemas complejos es bastante utilizado en evaluación del desempeño de sistemas computacionales [1]. La metodología que se propone para reproducir los niveles de desempeño en la interacción cliente/servidor, se basa en mediciones directas, en el control del tráfico en la red, y en la generación de modelos analíticos.

Las mediciones involucran la inserción de controladores de eventos en el sistema servidor de aplicaciones, tanto en nodos clientes como en nodos servidores. El control de tráfico presupone la posibilidad de generar patrones de carga tanto en la la red como en el servidor de aplicaciones, y que no existan distorsiones por carga externa (i.e. no controlada). Los modelos analíticos se aplican en sub-sistemas que son difíciles de medir directamente o bien que son simples de deducir por esa vía (p.ej. CSMA/CD). Gran parte de la metodología de evaluación requiere de la realización de mediciones en cada nivel de interacción relevante. Las mediciones deben entregar información que permitan identificar los diferentes componentes del tiempo de respuesta de la aplicación y el *overhead* que se produce por utilizar un esquema descentralizado a través de una red de comunicación.

Esta identificación se logra incorporando "relojes" tanto a nivel de cliente como de servidor que permitan conocer los retardos entre cada interfaz de comunicación, por ejemplo: a través del software de comunicación tanto del cliente como del servidor, a través del protocolo de comunicación de la red (p.ej. TCP/IP), en la transmisión de datos por la red, en el servicio local otorgado por el servidor, etc.

Dada la necesidad de acoplar mediciones desde diferentes nodos (cliente y servidor), en tiempo real, se debe sincronizar los relojes de cada uno, o en otras palabras conocer el desfase entre cada uno. Para ello, para iniciar una medición, se propone el siguiente algoritmo: (i) Se almacena la hora que registra el cliente ($t_1$) y se envía una señal al servidor, (ii) El servidor recibe la señal, y retorna de inmediato al cliente la hora de recepción ($t_2$) de dicha señal, medida en el servidor, (iii) El cliente recibe la respuesta y registra la hora de llegada ($t_3$). El desfase de los relojes se puede calcular como: $D = t_2 - (t_3 + t_1)/2$. El retardo para solicitar al sistema el tiempo transcurrido, a través de funciones del sistema operativo, es en general despreciable. Si no fuera así, este *overhead* debe ser incorporado al algoritmo de sincronización.

Para evitar resultados poco representativos, este procedimiento de sincronismo debe ser iterado un cierto número de veces, de manera de tener un desfase de tiempo promedio donde se minimicen las situaciones particulares al efectuar la medición. Así, es posible obtener, tanto para el cliente como para el servidor, un registro de los eventos medidos y los tiempos en los que ocurrieron. Finalmente estos registros pueden ser utilizados por un procedimiento que permita secuencializar los resultados para su interpretación.

Es posible identificar dos grupos de aplicaciones. El primero consiste de aplicaciones que no requieren interacción constante con el usuario (i.e. hay en un pedido inicial y su consiguiente respuesta desde el servidor). Estas aplicaciones son típicas en invocación de comandos remotos. El segundo grupo de aplicaciones, está formado por aquellas en las cuales existe una iteración del tipo solicitud/respuesta, entre el usuario y la aplicación (servidores de archivos, compiladores, sistemas de bases de datos etc.).

El desempeño de una red y de los sistemas componentes depende, además de los aspectos de configuración, del tráfico o carga involucrada. Utilizando el sistema servidor de aplicaciones mencionado anteriormente, se ha implantado un sistema generador de tráfico para incorporar distintos patrones de carga sobre la red en general, y sobre un nodo servidor en particular.

En cada nodo de la red hay un proceso denominado servidor de tráfico, que puede ser invocado desde cualquier cliente, recibiendo el patrón de tráfico a ser generado por el nodo donde se localiza este servidor. Así, el servidor de tráfico puede generar un número arbitrario de procesos clientes, cada uno conectado a un sub-proceso en el servidor de aplicaciones del nodo correspondiente. Mediante un monitor pasivo de red [13] es posible recolectar información del tráfico y retardos sobre la red, y de este modo incorporar los retardos del controlador de la red en los modelos de redes de espera que se propone generar (o bien validar los modelos analíticos de comportamiento de la red, en este caso Ethernet).

Con las mediciones se propone alimentar modelos del sistema, basados en modelos de redes de espera [9], para predecir resultados bajo diferentes valores de variables claves tales como: tipo de carga, velocidad de las CPUs, etc. Además, el sistema servidor de aplicaciones puede ser manipulado modificando elementos de configuración (cantidad de buffers, número de procesos dedicados, etc), para disponer de una mayor variedad de opciones en los modelos de proyección.

## V MODELACION DEL SISTEMA

En la construcción y uso de un modelo de red de espera [9], se incluyen, generalmente, tres etapas, a saber: i) Crear el modelo Base, ii) validar el modelo Base, y iii) utilizar el modelo para proyecciones futuras.

Para crear el modelo es necesario medir el sistema computacional, típicamente utilizando un monitor del sistema. Se debe efectuar una medición por un lapso de tiempo suficiente como para que incluya ya sea períodos de alta carga del sistema o alguna condición o proceso definido que se desea monitorear en forma exhaustiva. Debe incluirse en este monitoreo valores de las variables más relevantes del sistema, como uso de CPU y discos, número de ejecuciones de cada proceso (o clase de proceso), terminales activos, etc.

Deben ser definidos los componentes del modelo, identificando los centros de servicio que se desea contenga el modelo. Estos corresponden normalmente a la (o las ) CPU, los discos y de ser necesario un centro que represente la acción del controlador de red para representar la red de comunicación, entre otras cosas.

Deben se definidas las clases de procesos. Una clase de procesos consiste de "jobs" que tienen un origen común y que poseen consumos de servicio similares. Para cada clase de proceso se define el consumo promedio de una instancia del proceso en la CPU y en cada disco, en base a las mediciones del monitor. Finalmente, los datos entregados por el monitor se utilizan para dar valores a los parámetros de los componentes del modelo.

Para validar el modelo, los parámetros anteriormente identificados se pueden utilizar para alimentar un software de evaluación de redes de espera, por ejemplo MAP [12]. Las estimaciones de este software son comparadas con las mediciones del monitor y el modelo es validado o rechazado, caso en el que se debe volver a evaluar las clases y los valores de los parámetros. Luego, el modelo se puede usar para realizar proyecciones variando las características de algunos de sus componentes (p.ej. la velocidad de la CPU).

Al proyectar el comportamiento de cierta aplicación, un parámetro de utilidad es el tiempo de respuesta promedio al invocar la aplicación, bajo las condiciones "ambientales" de equipamiento y cantidad de usuarios (clientes) que están invocando la aplicación de forma concurrente. Otra variable importante, como parámetro de

entrada al modelo, es la demanda de CPU de la aplicación. Se considera una demanda por CPU "repartida", ya que hay requerimientos de CPU en los diferentes componentes del software cliente/servidor. Esta demanda por CPU puede ser utilizada para predecir la respuesta de la aplicación en otros ambientes, por ejemplo, si la velocidad de CPU del servidor se duplica.

Tanto el proceso cliente como el proceso servidor demandan un mínimo acceso a disco. Sin embargo, si la aplicación invocada es intensiva en lectura/escritura desde disco, las características de este dispositivo, como tiempo de "latencia" o búsqueda ("seek"), deben ser incorporadas como parámetros del modelo. En caso de existir mecanismos para almacenamiento temporal de información, como "cache disk" (caso típico de servidores de archivos) puede ser necesario incorporar esta información en el modelo.

### 5.1 Componentes del Modelo

Los componentes que debe incluir este modelo son básicamente los siguientes (ver fig. 3): (1) Procesos (o subprocesos) Clientes, (2) Procesos (o subprocesos) Servidores (3) Protocolos de Red. (p.ej. TCP/IP) (4) Red de comunicación (p.ej. Ethernet), (5) Aplicación invocada (p.ej.. sql, nfs).

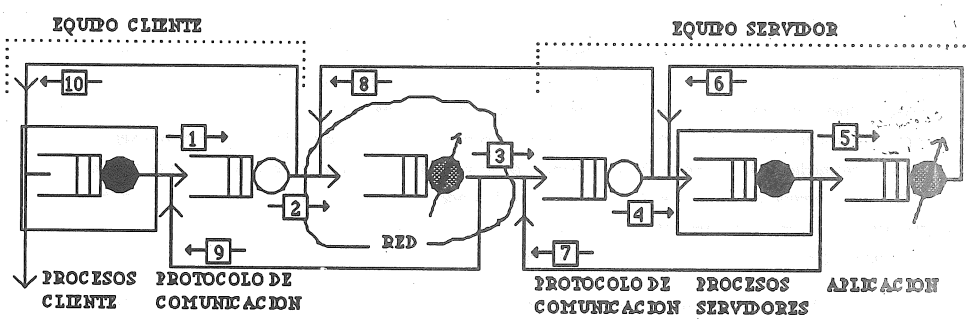


**Figura 3. Modelo de red de espera y flujo de datos**

Los "pasos" a representar en el modelo corresponden al flujo de datos entre los distintos componentes del sistema cliente/servidor. Dado que se requiere determinar el impacto de estos componentes en el tiempo de respuesta total, el flujo de datos incluye una transacción completa entre cliente y servidor (ver. fig. 3).

El paso 1 indica que un requerimiento (paquete) es enviado desde un cliente hacia la red, pasa por los protocolos de comunicación, en donde se le agrega un encabezado con las direcciones fuente, destino y otra información relevante para el protocolo. El paso 2 corresponde a la transmisión del paquete por la red, y el paso 3 al ingreso del paquete al protocolo de comunicación en el servidor. En el paso 4 el requerimiento es recibido por el proceso servidor y en el paso 5 la aplicación lo recibe y procesa. El mensaje de respuesta efectúa la secuencia inversa: enviado desde la aplicación hacia el subproceso servidor (paso 6), enviado al protocolo de comunicación del servidor (paso 7), transferido por la red (paso 8), procesado en el protocolo de comunicación del cliente (paso 9), y finalmente recibido por el proceso cliente y enviado al usuario (paso 10).

### 5.2 Procesos Cliente y Servidor

Para el proceso cliente se debe considerar tanto el tiempo de ejecución como la demanda de CPU que este proceso consume. El modelo debe reflejar las variaciones en los tiempos de respuesta, en función del número de clientes, n, el que se puede representar como una tasa de llegada. Para una aplicación, cada cliente pertenece a la misma clase de proceso; luego, si t es la demanda de servicio de un cliente, entonces $\lambda=n/t$ es la tasa de clientes/seg.

La invocación de la mayoría de las aplicaciones es efectuada mediante un requerimiento simple, a través de un paquete de datos. En este caso, la tasa de llegada es igual a la tasa de requerimientos por la red [paquetes/seg]. El tamaño de los paquetes enviados por la red es conocido por lo que se puede calcular cuanto aportan las aplicaciones a la carga de la red (paquetes/seg.* tamaño paquete / capacidad de transmisión). Teniendo acceso, además, a un monitor de redes [13] se puede conocer la carga total de la red, y obtener por lo tanto, el retardo a través de ella (ver 5.3).

El proceso servidor es similar al proceso cliente, existiendo un proceso servidor por cada proceso cliente remoto. Considerando el principio de conservación del flujo, la tasa de llegada de requerimientos al cliente es similar a la del servidor.

### 5.3 Modelo Representativo de la Red Ethernet

El desempeño de redes Ethernet ha sido extensivamente analizado [1,11,14]. Basado en la teoría de colas y en procesos "markovianos", se puede determinar el tiempo de servicio de un paquete en la red (retardo desde que un paquete es generado por el adaptador de la red hasta que llega al adaptador destino) en función del tráfico en la red. La carga es expresada como una proporción de la capacidad de transmisión de la red ($\rho$).

El tiempo se considera dividido en intervalos ("slots") cuya duración, S, es equivalente al tiempo de propagación ida-vuelta del canal. (tiempo requerido para que una colisión sea detectada por todas las estaciones). Si hay n estaciones activas (carga instantánea) y ninguna estación transmite, el "slot" es perdido; si sólo una estación transmite, adquiere el canal hasta que finalice de enviar el paquete. Si dos o más estaciones transmiten simultáneamente, ocurre una colisión y el intervalo es perdido. Si C es la capacidad de la red [bits/seg], y P el tamaño promedio de los paquetes [bits], el tiempo de servicio de un paquete, es igual al tiempo de transmisión (P/C) más el tiempo de contención, W. El primero no depende del tráfico, pero el último depende del número de paquetes en espera (carga instantánea), y es igual a la duración del *slot* multiplicado por el número de *slots* dedicados a contención ($W=S*(1-(1-1/n)^{n-1})/((1-1/n)^{n-1})$) [11].

Analizando la red como un modelo de Markov, la carga instantánea aumenta con la tasa de llegada de paquetes (cuyo valor medio es $\rho C/P$) y disminuye con la tasa de envío de los paquetes a destino. Esta es igual a capacidad de transmisión de la red, C/P, multiplicado por su eficiencia, E, cuando la carga instantánea es n, y es igual a $1/(1+ C/P*W)$. El tiempo medio de servicio de un paquete se obtiene (ecuación de Little), tomando el cuociente entre la carga instantánea promedio (número de paquetes en espera de servicio) y la tasa promedio de llegada de paquetes, $\rho$ C/P.

Para modelar esta red es posible utilizar este desarrollo para representarla como un Servidor de Flujo Equivalente (FESC) [9], dependiente de la carga ($\rho$) de la red y las características de la misma (p.ej. largo de la red, tamaño de los paquetes). Esta FESC puede ser validada mediante monitores de red, midiendo tiempos de respuesta ante una carga controlada.

### 5.4 Representación del Protocolo de Comunicación

El protocolo de comunicación corresponde a las capas medias de comunicación dentro de arquitecturas OSI [Open System Interconnection], y que comprenden básicamente a los niveles de red y transporte. La infraestructura de protocolos de comunicación utilizada en este estudio es TCP/IP.

Dentro del modelo, se requiere evaluar el tiempo de servicio de los paquetes en este protocolo. Ante un requerimiento por la red, el servidor responde al proceso cliente avisando su recepción. Registrando el tiempo en la máquina cliente al enviar el requerimiento, $T_1$, y al recibir la respuesta, $T_2$, se puede estimar el tiempo de servicio en el protocolo de comunicación. Con la carga de la red como entrada al FESC que la representa, se

obtiene el tiempo de transmisión, $T_3$, de un paquete. El tiempo del protocolo de comunicación, $T_C$, se estima como $((T_2-T_1)-T_3)/4$, dado que tanto el paquete de solicitud como el de respuesta "entra y sale" del protocolo.

### 5.5 Representación de la aplicación

La aplicación invocada puede representar en sí un modelo computacional relativamente complejo, que incluye, entre otros, contención por memoria, CPU y discos. Este es el caso de aplicaciones que proveen servicio de archivos o bases de datos. En general, la aplicación se puede representar por medio de un modelo abierto [9] (transaccional), representando cada componente crítico mediante centros de servicio, con una tasa de llegada que depende del número de clientes en acceso concurrente a la aplicación.

Considerando la provisión de "relojes" de control entre los distintas interfaces de comunicación, se pueden medir los tiempos de respuesta desde la aplicación, y compararlos con los tiempos que entrega el modelo. Dada la infraestructura cliente/servidor disponible, el proceso de medición se puede repetir en distintas máquinas servidoras, lo cual puede servir como un elemento adicional para la validación del modelo de la aplicación.

Cuando no sea posible medir exhaustivamente los centros de servicio de la aplicación, ésta se puede representar mediante una FESC, considerando el tiempo de respuesta global en función del número de clientes activos. La FESC debe representar variaciones de parámetros del equipo servidor (p.ej. velocidad de la CPU). Luego, es necesario cuantificar el retardo de la aplicación, en función del número de clientes concurrentes.

Para el proceso servidor, cada nueva solicitud es atendida por un subproceso que activa a la aplicación, y es independiente de la máquina a que responde. Por lo tanto, utilizar m máquinas cliente sólo aumenta la complejidad de las mediciones. Una ventaja de usar m máquinas cliente es aumentar la carga de la red para cuantificar la contención ante un aumento de tráfico, pero el contar con un generador de tráfico hace esto innecesario.

## VI MEDICIONES PRELIMINARES

A continuación se presentan mediciones realizadas en el ambiente cliente/servidor descrito anteriormente. Estas mediciones pretenden mostrar parte de la metodología de evaluación de desempeño propuesta en este trabajo, y presentar una tendencia en el desempeño de distintas etapas involucradas en las transacciones cliente-servidor.

Se incluyó una máquina servidor (UNISYS U6050/UNIX V.3) y una máquina cliente (VAX 3400/ULTRIX-32 V3.1), usando como aplicación remota el sistema de base de datos SQLPLUS, y efectuando los clientes consultas del tipo "help select" (que entrega 6 Kbytes de información). En este caso no se incorporó carga artificial sobre la red, sino que sólo se varió el número de clientes concurrentes (de 1 a 10), percibiéndose una carga sobre la red que fluctuaba entre un 1% y un 3% de su capacidad. Los tiempos que se consideraron son:

- Conexión: tiempo que requiere el cliente para conectar su canal de comunicación con el servidor.
- Inicio aplicación: tiempo en que el proceso servidor crea directorios temporales, invoca a la aplicación, y crea un canal entre ésta y su salida (*output*).
- Aplicación: El tiempo que demora la aplicación para satisfacer el requerimiento y enviar la respuesta al cliente (este tiempo se deglosa en distintos niveles de desempeño en las transacciones cliente/servidor).

La fig. 4 muestra estos tres componentes de tiempo. Se aprecia que el "overhead" de iniciar una conexión remota es bajo comparado con el tiempo para satisfacer el requerimiento remoto. Al crecer el número de clientes, el tiempo de conexión aumenta levemente. El tiempo de inicio de la aplicación involucra la activación de un mayor número de procesos y la creación de áreas de trabajo temporales, y en este caso este tiempo es más sensible al aumento en el número de clientes.

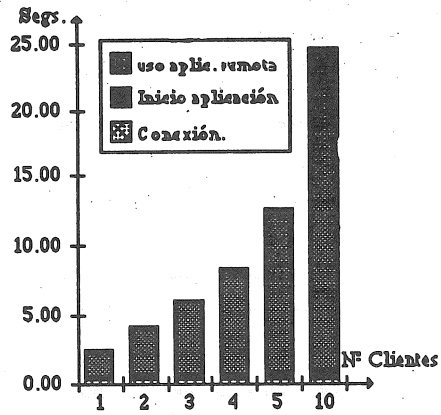


**Figura 4: tiempos de respuesta**

### 6.1 Deglose de tiempos en acceso a la aplicación remota

Dentro del tiempo de respuesta de la aplicación se incluye el tiempo que requieren los paquetes de datos en ser enviados desde la aplicación hacia el cliente. Aquí pueden ocurrir dos situaciones de retardo: espera del servidor por datos desde aplicación, y envío de los datos por la red y recepción de confirmación desde el cliente. Cuando hay espera de datos desde la aplicación, la red no produce contención.

Para estimar la proporción de tiempo en espera por la aplicación, se midió tanto el número de lecturas nulas desde la aplicación (al intentar leer desde la aplicación aún no hay datos) y el número de lecturas exitosas desde la aplicación (ver fig.5) Se puede apreciar que la congestión producida por la aplicación es alta.

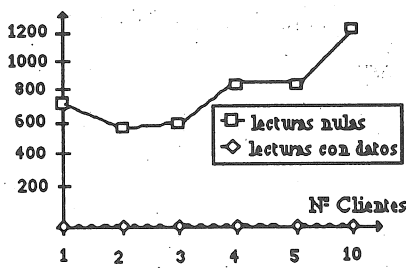


**Figura5 Lecturas desde la aplicación**

La fig. 6 muestra el tiempo de contención en la red. Se aprecia que este retardo es poco significativo dentro del tiempo total (ver fig. 4). El incremento en este tiempo es debido al aumento de "overhead" involucrado en el protocolo de comunicación (TCP/IP), dado que la carga en la red se mantuvo en un mínimo.

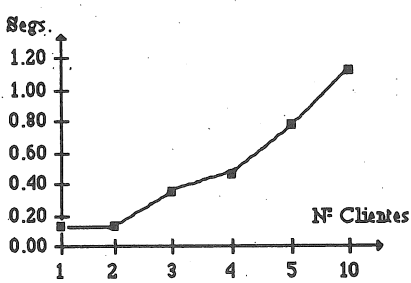


**Figura 6. Contención en sistema de comunicación**

Finalmente, en la fig. 7 se muestra la demanda de CPU requerida por cada proceso, en función de la demanda (en número de clientes). Cabe señalar que, por cada cliente existe un proceso en la máquina cliente y dos procesos en la máquina servidora. Esta demanda depende de la velocidad de la CPU tanto en el cliente como en el servidor.

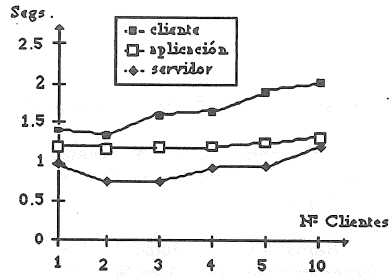


**Figura7: Demandas de CPU por proceso**

Mediciones como las anteriores se pueden realizar en distintos escenarios (tipos de computadores clientes y servidores, tráfico en la red, etc), y con diferentes aplicaciones remotas. Los datos identificados muestran distintos niveles de desempeño de aplicaciones cliente/servidor y se pueden incorporar a modelos de redes de espera como los descritos anteriormente.

## VII. CONCLUSIONES

En este trabajo se propone una metodología para evaluar el impacto de los diferentes niveles de desempeño de una red, en aplicaciones cliente-servidor. Se realizaron herramientas de comunicación, para generar patrones de carga reales en una red y para medir el desempeño de los distintos elementos de interacción en las transacciones entre clientes y aplicaciones remotas.

Se ha definido un modelo general del sistema que, una vez llevada a cabo mediciones, bajo distintos escenarios de carga y configuraciones, permita predecir el impacto de los distintos niveles de comunicación, y definir cotas y patrones de desempeño en diversas aplicaciones de red.

Finalmente, se presenta un ejemplo de utilización de la infraestructura de evaluación desarrollada, que entrega resultados de desempeño en el acceso a una aplicación remota. A futuro, se espera crear una plataforma automatizada para la experimentación con la metodología de evaluación propuesta.

BIBLIOGRAFIA


[1] G. Almes and E. Lazowska. "The behavior of Ethernet-like Computer Communications Networks". Proceedings of the 7th Symposium on Operating Systems Principles, 66-81. 1979.

[2] F. Aurtenechea, I. Casas, M. Kutulas. "Diseño y Evaluación de un Servidor de Procesos para una Red Local". XVII Conferencia Latinoamericana de Informática", CLEI- Panel'91 Expodata, Caracas, Venezuela, 459-476 Julio 1991.

[3] G. Bernard G, A. Duda, Y. Haddad, G. Harrus. "Primitives for Distributed Computing in a Heterogeneous Local Area Network Environment", IEEE Transactions on Software Engineering, 1567-1578, Vol 15, No. 12, Dec 1989.

[4] P.I. Boulton, R.E. Soper, E.S Lee. "Simulation of Two LANs". Technical Report CSRI-233, University of Toronto. March 1990.

[5] W. Bux. "Performance issues in local-area networks". IBM Systems Journal, Vol 23, Nº 4, 351-374, 1984.

[6] D. Cheriton. "The V Distributed System". Communication of the ACM, Vol 31, No. 3, 314-333, March 1988.

[7] T. Gonsalvez. "Performance Characteristics of 2 Ethernets: an Experimental Study", ACM Sigmetrics, Performance Evaluation Review, Vol 13 Nº 2, August 1985.

[8] D. Jacobson, S. Gaitonde, J. Kim, J. Lee, D. Rover, M. Sarwar, M. Shafiq, "A Master/Slave Monitor Measurement Technique for an Operating Ethernet Network", IEEE Network Vol. 1, Nº 3, July 1987.

[9] E. Lazowska, J. Zahorjan, G.S. Graham, K.C. Sevcik, "Quantitative System Performance Computer System Analisys Using Queueing Network Models", Prentice Hall, New Jersey, 1984.

[10] E. Lazowska, J. Zahorjan, D. Cheriton, W. Zwaenepoel. "File Access Performance of Diskless Workstations". ACM Transactions on Computer Systems. Vol. 4 Nº 3, 238-268, August 1986.

[11] R. Metcalfe, R. Boggs "Ethernet: Distributed Packet Switching for Local Computer Networks". Communicaction of teh ACM, Vol. 19, Nº 7, 395-404, 1976.

[12] Quantitative System Performance Inc., "MAP - Release 1.2 User's Guide & Reference Guide", Seattle, Washington, 1985.

[13] K. Shah. "Managing networks of the '90s". Data Communications International, 95-106, December 1989.

[14] J.F. Shoch, J.A. Hupp. "Measured Performance of an Ethernet Local Network" CACM, 23:12, pp. 711-721, Dec. 1980.

[15] A. Tanenbaum. & R. Van Renesse. "Distributed Operating Systems". Computing Surveys, Vol. 17, Nº 4, 419-470, 1985.