Adaptación de ns-2 para una Variante del Protocolo 802.11e (EDCA)

Guillermo Reggiani[1] , Omar Alimenti[1] , Guillermo Friedrich[1], Santiago Tonietti[1], Federico Maidana[1], Damian Gomez de Marco[1],

[1] UTN Facultad Regional Bahía Blanca, Departamento de Electronica, 11 de Abril 461, (8000), Bahía Blanca
{iealimen}@criba.edu.ar, {gfried, ghreggiani}@frbb.utn.edu.ar, {stonietti, damiangomezdemarco, maidana.fede}@gmail.com

Resumen. En el estudio de los diversos mecanismos tendientes a mejorar los protocolos de redes de datos, se requiere de simuladores para evaluar su comportamiento. El simulador Network Simulator ns-2 [1] es uno de los referentes utilizados en los trabajos de investigación sobre las redes de datos inalámbricas. En este trabajo se muestra la forma de adaptar el código (abierto) del ns-2 para poder evaluar redes ad-hoc. A modo de ejemplo se implementó una modificación del código del simulador para analizar una variante sobre el mecanismo EDCA (Enhanced Distributed Channel Access) del estándar IEEE 802.11e [2]. Esta propuesta denominada RT-EDCA [3]. (Real Time EDCA) permite garantizar la latencia de los mensajes para sistemas de tiempo real sobre el Control de Acceso al Medio (MAC) de EDCA.

Palabras claves: ns-2, EDCA, simulación, algoritmo de backoff, Ventana de Contención.

1 Introducción

Una herramienta muy útil dentro del estudio de la ingeniería radica en la utilización de simuladores, que permite comprender de mejor manera los fenómenos que se llevan a lugar en la aplicación de la teoría. Este es el caso del software ns-2 (Network Simulator 2), que es una de las herramientas más potentes en lo que a simulación de sistemas de comunicaciones inalámbricas se refiere; no solo brinda la ventaja de ser un software de libre distribución sino que consta de una amplia documentación para su uso.

Entre los usos más habituales que se le puede dar a este tipo de simuladores se encuentran:

- Simular estructuras y protocolos de redes de todo tipo (satélite, inalámbricas, cabledas, etc).
- Desarrollar nuevos protocolos y algoritmos y comprobar su funcionamiento
- Comparar distintos protocolos en cuanto a prestaciones.

Además es un instrumento muy flexible ya que da la posibilidad de trabajar con scripts tcl que permiten agregar toda la potencia de un lenguaje de programación a los propios elementos de simulación.

El propósito del presente trabajo es mostrar la forma de modificar el código abierto de los distintos módulos del simulador como también realizar cambios en sus parámetros, adecuándolo a éste a distintos modelos de protocolos de redes de datos para poder evaluar sus funcionamientos a través de las simulaciones.

A modo de ejemplo se tomó la propuesta de [3] y en base a esta se efectuó la adaptación del simulador tanto a nivel de código como de sus parámetros. La misma es una variante del mecanismo de Control de Acceso al Medio EDCA del estándar IEEE 802.11e para sistemas de tiempo real denominado RT- EDCA (Real Time EDCA).

El trabajo se organiza como sigue: la sección 2 provee una descripción del mecanismo RT-EDCA utilizado como ejemplo, en la sección 3 se explica la adaptación del simulador ns-2, en la sección 4 se realizan dos simulaciones sobre el modelo de ejemplo. A continuación en la sección 5 aparecen las conclusiones y por último las referencias.

2 RT-EDCA: Optimización de la capa MAC

EDCA controla el acceso al canal inalámbrico en base a un mecanismo (Fig. 1) diferenciado de QoS con cuatro AC: AC_BK (Background) para niveles de prioridad más bajos (1 y 2), AC_BE (Best Effort) para los siguientes (0 y 3), AC_VI (Video) para 4 y 5 y AC_VO (Voice) para las más altas (6 y 7). De acuerdo a su prioridad, una trama será ubicada en alguna de esas cuatro categorías. Cada AC ejecuta un proceso de tiempo de espera (backoff) independiente, denominado ventana de retroceso (BW: backoff window) para determinar el instante de inicio de la transmisión de sus tramas. El proceso de BW para cada AC, está definido por cuatro parámetros configurables: Espacio de Separación Entre Tramas para Arbitraje (AIFS), CWmín, CWmáx y TXOPlímite [4] [5].

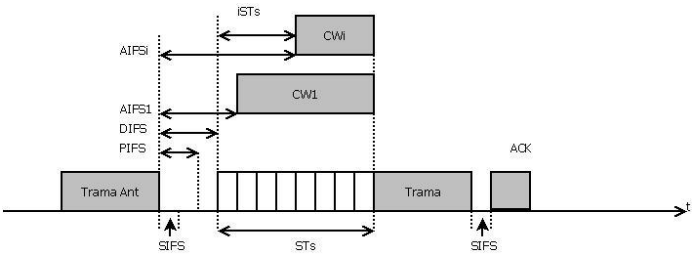

Fig. 1. Esquema básico EDCA.

El objetivo básico propuesto es configurar los parámetros de EDCA [6] para eliminar los factores probabilísticos y asegurar así una latencia máxima de transmisión [7][8], a fin de que sea apto para tráfico de tiempo real. El nuevo

protocolo, denominado RT-EDCA, está basado en las siguientes pautas y condiciones de funcionamiento:

- A cada tipo de trama le corresponde una determinada prioridad, conocida desde el instante inicial y distinta a cualquier otra (a la manera del bus CAN) [9].
- La prioridad se indica mediante un valor numérico comprendido entre cero para la máxima y un cierto número positivo N para la mínima (N depende del número de tipos de mensajes que se van a manejar en el contexto de la aplicación).
- En caso de requerimientos simultáneos, se debe transmitir el de mayor prioridad.
- Todas las estaciones transmisoras (STA) son capaces de escucharse entre sí (no existen nodos ocultos).
- La lógica del mecanismo MAC debe evitar la ocurrencia de colisiones.
- Todo el tráfico es RT-EDCA (no se admite tráfico mixto con EDCA y/o DCF).
- La ausencia de ACK implica que se produjo una colisión o bien que la transmisión se alteró por ruido o interferencia electromagnética.

RT-EDCA introduce las siguientes variantes al protocolo EDCA:

- Se fija CWmín = CWmáx = 0, por lo que BW=0.
- Se establece un tiempo de arbitraje AIFSi distinto para cada tipo de trama, tal que a menor AIFSi la prioridad es mayor.

La Fig. 2 presenta los aspectos básicos de RT-EDCA. Una STA espera que el medio permanezca inactivo durante AIFSi y luego inicia la transmisión. Si durante la espera el medio es ocupado, a diferencia de EDCA, el intento se aborta para reiniciarse cuando el medio vuelva a quedar ocioso.

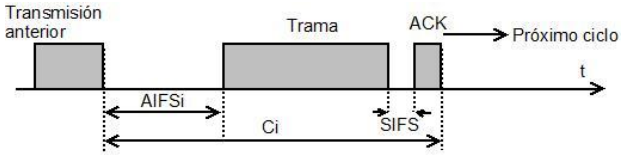

Fig. 2. RT-EDCA: Esquema básico.

3 Adaptación del simulador ns-2

El simulador ns-2 es el más utilizado para simulaciones de distintos algoritmos implementados sobre redes LAN inalámbricas bajo el estándar 802.11. ns es un simulador orientado a objetos escrito en C++ que utiliza como interface con el usuario el intérprete OTcl .

En nuestro caso se tomó una adaptación de este el cual incorpora la actualización del estándar [10] para la simulación del Control de Acceso al Medio - MAC 802.11e.

La implementación completa de EDCA está dividida en cuatro módulos básicos. Cada módulo contiene todas las funciones y parámetros necesarios para poder llevar a cabo en forma satisfactoria, una simulación del Control de Acceso al Medio. Los módulos antes mencionados, se pueden identificar como:

- Drop Tail
- Priority Queue
- MAC 802.11e Timers
- MAC 802.11e

Drop Tail: es el módulo encargado de tomar las decisiones para desechar los paquetes encolados, que no han podido ser enviados. Se basa en parámetros como el TXOPlimit (Tiempo límite de transmisión de un paquete) y la longitud que se haya definida para la cola.

Priority Queue: es el módulo en donde están definidos los parámetros necesarios para el control de acceso al medio, según la prioridad requerida para el tráfico. Se tienen cuatro parámetros por prioridad:

- CW_min, CW_max: valor mínimo y máximo que pueden tomar las ventanas de contención.
- AIFSN: número por el que se multiplica una ranura de tiempo (slot time) para poder calcular el espacio de arbitraje entre tramas (Arbitration Inter-Frame Space).
- TXOPlimit: Tiempo límite de transmisión de un paquete.

La tabla 1 muestra un resumen de los valores que tiene cada uno de los parámetros para las distintas clases en las que puede operar en EDCA.

Table 1. Parámetros por defecto para cada clase EDCA.

AC	CWmin	CWmax	AIFSN	MaxTXOP
Background (AC_BK)	31	1023	7	0
Best Effort (AC_BE)	31	1023	3	0
Video (AC_VI)	15	31	2	3.008 ms
Voice (AC_VO)	7	15	2	1.504 ms

Este módulo tiene un papel importante para poder llevar a cabo las modificaciones en el estándar EDCA, y lograr simular RT-EDCA.

MAC 802.11e Timers: es el módulo encargado de hacer el manejo de los temporizadores de todos los tráficos que se generen en la simulación. Maneja los tiempos de IFS (Inter-Frame Space) y la cuenta del Backoff. Además este módulo se encarga de calcular la ventana de contención en caso de que haya sucedido una colisión en el envío de un paquete o bien se haya perdido el ACK (Acknowledge), para ello usa como parámetros CW_min, CW_max, AIFSN y el Tiempo de Ranura. Sobre éste punto también se realizaron modificaciones para poder implementar RT-EDCA.

802.11e: este módulo es aquel que se encarga de armar los paquetes, mandarlos a la cola, verificar el estado del medio, verificar el ACK, generar los envíos de los paquetes y la recepción de los mismos. Para ello utiliza todos los parámetros definidos en la norma IEEE802.11e, además se apoya en los tres módulos mencionados anteriormente.

3.1 Implementación de RT-EDCA

El proceso de simulación se realiza a través del desarrollo de scripts en OTcl, en el cual se especifican protocolos, estructuras físicas, tráficos y orden de los eventos, entre otros aspectos a simular. Luego viene el procesamiento interno del archivo dado y los resultados de la simulación pueden ser entregados en tres tipos de archivos distintos: los archivos generados por el usuario que son del formato y necesidades definidas por éste (*salida.tr*), los archivos de traza que muestran información de los paquetes que están viajando en la red y los archivos .nam para ser visualizada por Nam. Esta última es una herramienta de animación para visualizar la simulación de la red.

Descripción y configuración de la topología a simular. La conformación de la red va a estar dada por estaciones inalámbricas que transmitirán datos con calidad de servicio según lo especificado en el estándar 802.11e a una tasa de transferencia de 11 Mbps. La cantidad de estaciones o nodos es variable. La Fig. 3 muestra una topología con 5 nodos.

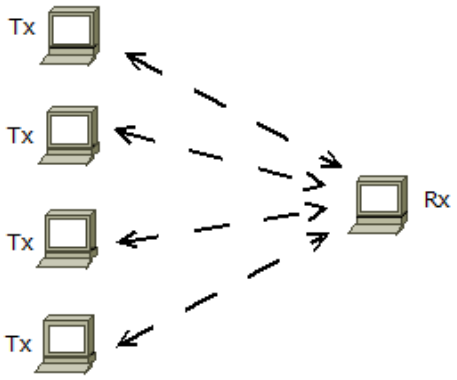

Fig. 3. Topología de red

Los parámetros inalámbricos para los nodos se definen dentro del script de la simulación de la siguiente manera:

```
set opt(chan) Channel/WirelessChannel ;# channel type
set opt(prop) Propagation/TwoRayGround ;# radio-
propagation model
```

```
set opt(netif)Phy/WirelessPhy ;# network interface type
set opt(mac)  Mac/802_11e      ;# MAC type
set opt(ifq)  Queue/DTail/PriQ  ;# interface queue type
set opt(ll)   LL                   ;# link layer type
set opt(ant)  Antenna/OmniAntenna  ;# antenna model
set opt(ifqlen) 50                 ;# max packet in ifq
set opt(adhocRouting)   DSR     ;# routing protocol DSR
```

Dentro de los parámetros importantes se observa el seteo del tipo de MAC (Mac/802.11e) y el tipo de encolamiento (Queue/DTail/PriQ). Se utiliza el modelo de encolamiento DTail, el cual utiliza cuatro buffers de diferente prioridad, en donde cada uno de ellos maneja su propio tráfico y algoritmo de backoff. La longitud de cada uno de los buffers es de 50 paquetes (ifqlen), esto significa que los paquetes excedentes serán descartados.

Otro punto a configurar dentro del script .tcl [11] de la simulación es el tipo de tráfico a transmitir. El tráfico simulado fue CBR (Constant Bit Rate) sobre un protocolo de transporte UDP (User Datagram Protocol).

Para poder implementar RT-EDCA en el simulador, nos basamos en la implementación del protocolo 802.11e. Se hicieron diversos análisis sobre el código de cada módulo y la interacción entre ellos, con un seguimiento del uso de las siguientes variables:

- CW_min
- CW_max
- AIFSN

Estas variables tienen un rol importante en las reformas planteadas al estándar EDCA. Como primera medida se modificaron las variables de tal forma que cumplieran con el modelo RT-EDCA.

El módulo *Priority Queue*, cuenta con un archivo .tcl que es donde se definen los valores de CW_min, CW_max y AIFSN para cada una de las clases de prioridad del estándar EDCA. A continuación se muestra parte del archivo *priority.tcl*:

```
# parameters for Queue 0
    $ifq Prio 0 PF 2
    $ifq Prio 0 AIFS 2
    $ifq Prio 0 CW_MIN 7           ;# (aCWmin+1)/4 - 1
    $ifq Prio 0 CW_MAX 15          ;# (aCWmin+1)/2 - 1
    $ifq Prio 0 TXOPLimit 0.003264
```

Este es el caso de los parámetros de prioridad 0 (mayor prioridad) donde puede observarse los valores de CW y AIFS y TXOPLimit. El archivo original contiene las 4 clases de prioridades indicadas en el estándar.

Para RT-EDCA, se crearon más clases de prioridad, tantas como mensajes a simular. A cada clase se le asignó un valor de AIFSN distinto, siendo el más chico (AIFSN=2) para la clase de mayor prioridad y se incrementó el valor en uno a medida que bajaba el nivel de prioridad.

Por otro lado, las variables CW_min y CW_max, se igualaron a cero, de tal manera que el conjunto de parámetros para un ejemplo de 5 clases de prioridades sería como se muestra en la tabla 2:

Tabla 2. Parámetros para cada clase RT-EDCA para una red con 5 nodos.

AC	CWmin	CWmax	AIFSN
0	0	0	2
1	0	0	3
2	0	0	4
3	0	0	5
4	0	0	6

Al hacer los valores de CW_min y CW_max cero, se elimina la aleatoriedad que produce el tamaño de la ventana de contención, con lo cual el valor de espera que van a usar los nodos a la hora de competir por el medio será solo el AIFS. Los valores de CW en cero generan una inconsistencia en la simulación, por tal motivo se modificó el archivo mac-timers_802_11e.cc. La variable *rtime* dentro del proceso de backoff toma una valor aleatorio comprendido entre el valor 0 y CW_min, la misma se fijó en 0 logrando así que cada nodo tenga un tiempo de espera distinto (dado por los AIFS) para competir por el medio.

4 Evaluación de Desempeño

Para la evaluación del desempeño del modelo RT-EDCA se utilizaron mensajes periódicos de pequeño tamaño que por lo general se utilizan en sistemas de control sobre redes inalámbricas (WNCS) [12].

Se generaron los scripts de simulación .tcl para los dos escenarios a evaluar definiendo en el mismo el tiempo de simulación, parámetros capa física, tipo de MAC, topología, tráfico, carga útil (payload), entre los más importantes parámetros. Se adoptó para ambos escenarios una tasa de transferencia a 11 Mbps, preámbulo largo (192 μs), carga útil de 50 y 500 bytes (más 36 bytes de encabezado) y ACK de 14 bytes. A continuación se muestra parte del código del script de la simulación donde se definen los parámetros antes mencionados:

```
set num_total_nodos 41    ;# número de nodos
set tamano 50             ;#tamaño de los paquetes
.
set opt(netif) Phy/WirelessPhy ;# network interface
set opt(mac)   Mac/802_11e     ;# MAC type
set opt(ifq) Queue/DTail/PriQ ;# interface queue type
.
#Creación de Tráfico
proc agentes-traficos {nodo tamano intervalo receptor
prioridad} {
  set ns [Simulator instance]
```

```
    set emisor [new Agent/UDP]
    $emisor set prio_ $prioridad
    $emisor set class_ $prioridad
    $ns attach-agent $nodo $emisor
    set cbr [new Application/Traffic/CBR]
    $cbr set packetSize_ $tamano
    $cbr set interval_ $intervalo
    $cbr attach-agent $emisor
    $ns connect $emisor $receptor
    return $cbr
}
```

En el primer ensayo se considera un conjunto de 40 nodos que transmiten mensajes periódicos de igual tamaño y período. Partiendo de un instante crítico, se evaluó mediante simulaciones el tiempo requerido para completar la última transmisión. El resultado obtenido es el mínimo período que debería asignarse a cada uno de los mensajes de este conjunto

En el simulador, al margen del cambio efectuado en el código del archivo *mac-timers_802_11e.cc* para RT-EDCA, se modificó el archivo *priority.tcl* incorporando las prioridades para cada uno de los nodos, definiendo para el caso de mayor prioridad (0) AIFS = 2, CW_min = 0, CW_max = 0 y TXOPLimit = 0. El AIFS se incrementa de a 1 para cada una de las prioridades, el resto de los parámetros siguen siendo 0. El tiempo de ranura es ST = 20 μseg. El tiempo de generación de los mensajes en los nodos se definió en 100 mseg, permitiendo obtener el tiempo mínimo que se necesita para que todos los mensajes puedan ser transmitidos.

En la Fig. 4 se presentan los resultados obtenidos en las simulaciones, tanto para RT-EDCA como para EDCA estándar. Se observa que RT-EDCA presenta un mejor desempeño hasta 28 nodos (con paquetes de 50 bytes) y hasta 40 nodos (con paquetes de 500 bytes). A partir de allí es superado por EDCA, debido a que los valores crecientes de AIFS aumentan la duración del ciclo de transmisión de cada trama.

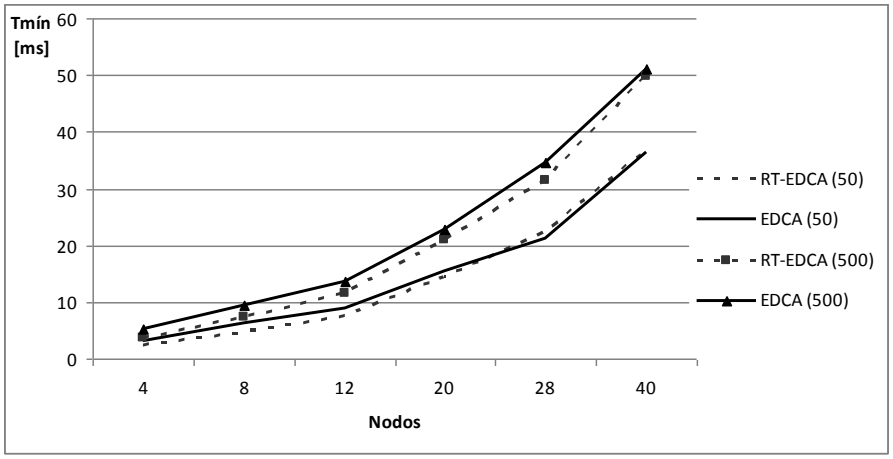

Fig. 4. Período mínimo comparativo entre RT-EDCA vs EDCA

En el siguiente ensayo, cuyos resultados se observan en la Fig. 5, se evaluó el desempeño de una red con 40 mensajes de 50 bytes de carga útil, usando cuatro esquemas de agrupamiento de mensajes en clases, tanto para EDCA como para RT-EDCA. El objetivo fue determinar el período mínimo entre transmisiones que podría asignarse a dicho conjunto de mensajes. En todos los casos se considera que cada nodo se encarga de una clase de mensajes. Las opciones de agrupamiento fueron a razón de un mensaje por nodo o clase (o sea: sin agrupamiento), dos mensajes por nodo (20 nodos), cuatro mensajes por nodo (10 nodos) y ocho mensajes por nodo (5 nodos).

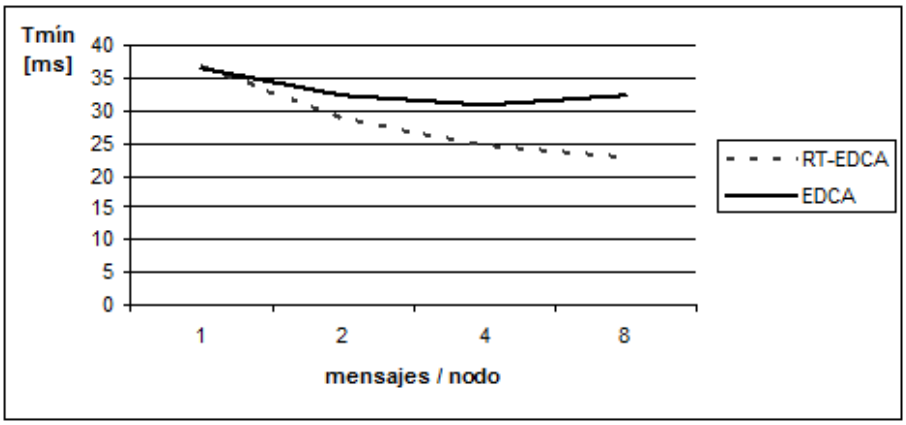

Fig. 5. Período mínimo para 40 mensajes con diferentes agrupamientos

En el simulador para RT-EDCA se modificó de forma análoga a lo explicado en el anterior ensayo el archivo *priority.tcl* y en el script de la simulación se agregó una rutina que asigna los mensajes a los nodos dependiendo de cuál sea la agrupación.

En el caso de EDCA, los mensajes se distribuyeron equitativamente entre las cuatro AC.

5 Conclusión

Este trabajo tiene como objetivo mostrar la forma en que se lleva a cabo la adaptación del simulador ns-2 para poder realizar la evaluación de las distintas propuestas tendientes a optimizar los protocoles de redes de datos.

Se realizó la modificación del código y sus parámetros en base a un modelo tomado como ejemplo el cual plantea una mejora en tiempo real sobre el mecanismo MAC EDCA del estándar 802.11e. En las evaluaciones efectuadas a través de las simulaciones se tomó como parámetro para el análisis el periodo mínimo de tiempo que necesitan los nodos para transmitir los mensajes. Se evaluaron distintos escenarios con distintas cantidades de nodos.

Probablemente ns-2 es el simulador de redes de código abierto más extendido tanto en investigación como para propósitos docentes. Debido a que es un simulador

consolidado, resulta una herramienta de suma importancia para los investigadores a los cuales les facilita la evaluación de los mecanismos elaborados para la optimización de distintos protocolos y estándares en el ámbito de las redes de datos.

Referencias

1. The Network Simulator- ns-2, <http://www.isi.edu/nsnam/ns/index.html/>, April 28, 2007.
2. "IEEE Std 802.11; Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications", 1999, actualización: Junio (2007).
3. Alimenti O., Friedrich G., Reggiani G., Tonietti S., Maidana F. y Gomez de Marco D., "Comunicaciones en Tiempo Real adecuando el Protocolo 802.11e (EDCA)"; Simposio Argentino de Tecnología – 40 Jornadas Argentinas de Informática e Investigación Operativa – JAIIO 2011, Córdoba, Argentina, 29/08 al 02/09 de 2011.
4. "IEEE Std 802.11e; Part 11: Wireless LAN MAC and Physical Layer (PHY) Specifications and Amendment 8: MAC Quality of Service Enhancements" (2005).
5. Willig A., "Recent and Emerging Topics in Wireless Industrial Communications: A Selection", IEEE Transactions On Industrial Informatics, Vol. 4, Nº. 2, May (2008).
6. Serrano P., Banchs A., Patras P. and Azcorra A., "Optimal Configuration of 802.11e EDCA for Real-Time and Data Traffic". IEEE Transactions on Vehicular Technology, Vol. 59, No. 5, June 2010, pp. 2511-2528.
7. Pereira da Silva M. and Becker Westphall C., "Performance Analysis and Service Differentiation in the MAC SubLayer of IEEE 802.11e Ad Hoc Networks", Proceedings of the Advanced Industrial Conference on Telecommunications, IEEE (2005).
8. Alimenti O., Friedrich G. and Reggiani G., "A Free-Collision MAC Proposal for 802.11 Networks", 28th Edition (SBRC 2010) and 12th Brazilian (WTR 2010), Gramado, ISSN: 2177-496X, pp: 89 – 100 (cd), May 24 – 28, 2010, Brasil.
9. CAN Specification 2.0, Robert Bosch GmbH, can2spec.pdf (1991).
10. Wietholter S., Hoene C.,"Design and Verification of an IEEE 802.11e EDCA", Technical Report TKN-03-19, November 2003, Berlin
11. http://www.frbb.utn.edu.ar/electronica/sitic/
12. Li Gui, Yu-Chu Tian, Colin Fidge. (2007) "Performance Evaluation of IEEE 802.11 Wireless Networks for Real-time Networked Control Systems". Proc. of The 2007 International Conference on Embedded Systems and Applications, Las Vegas, USA.