INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE INGENIERÍA MECÁNICA Y ELÉCTRICA

EVALUACIÓN DEL DESEMPEÑO DEL PROTOCOLO ALOHA

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

INGENIERO EN COMUNICACIONES Y ELECTRÓNICA

PRESENTAN

CESAR ALEJANDRO CARREÓN GUERRERO

VÍCTOR JOSÉ GATICA ACEVEDO

ASESOR: ERIC GÓMEZ GÓMEZ

MÉXICO, D.F. NOVIEMBRE 2008

AGRADECIMIENTOS

A mi papá José Isabel Gatica Riachi, por haberme encaminado siempre a la superación personal y aunque ya no estés presente físicamente todos tus sabios consejos están reflejados en mis éxitos.

A mi mamá María del Rocío Acevedo Tornéz, por el amor que me has dado y apoyo en mi formación profesional.

A mis hermanas Miriam Janet, Isabel Rocío, Karen Christally y a mi hermano Jesús Emmanuel, por darme momentos gratos en la vida.

A mis amigos Cesar Alejandro Carreón Guerrero y Carlos Antonio Galindo Virues.

A todos mis familiares y amigos.

A todos los profesores de la E.S.I.M.E por abrirnos las puertas al conocimiento, al IPN por darme la oportunidad de formar parte de esta gran institución y a nuestro asesor Eric Gómez Gómez por su apoyo en la realización de este trabajo..

Víctor José Gatica Acevedo

AGRADECIMIENTOS

A mi mamá Maria del Pilar Carreón Guerrero, por todo el apoyo, amor y comprensión que me has brindado, que me ha ayudó a seguir adelante y así poder concluir un ciclo más.

A mi abuelita Ofelia Guerrero Martínez, cuyos cuidados y consejos hicieron de mí una mejor persona.

A mi abuelito Martín Carreón Parada por haber sido como un padre para mí.

A mis tíos y tías que de una u otra forma me han impulsado en mi desarrollo profesional.

A mi amigo Víctor José Gatica Acevedo quien ha sido un gran amigo y sin el cual no hubiera podido desarrollar este trabajo.

A mis amigos Miguel, Eduardo, Dayna y Armando por su amistad incondicional y que siempre fueron apoyo en momentos difíciles.

A todos los profesores de la E.S.I.M.E por abrirnos las puertas al conocimiento,al IPN por darme la oportunidad de formar parte de esta gran institución y a nuestro asesor Eric Gómez Gómez por su apoyo en la realización de este trabajo.

Cesar Alejandro Carreón Guerrero

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

OBJETIVO GENERAL

En este trabajo de tesis se propone como objetivo general la evaluación del desempeño del protocolo de acceso múltiple Aloha en un escenario satelital mediante simulación.

OBJETIVOS ESPECÍFICOS

Seleccionar el simulador adecuado para la evaluación del desempeño de redes satelitales.

Desarrollar un escenario satelital adecuado para la realización de la evaluación del desempeño.

Buscar el máximo rendimiento para la red con diferentes parámetros de desempeño.

JUSTIFICACIÓN

En los últimos años, los sistemas de comunicaciones se han ido desarrollando, tendiendo a la convergencia de datos, voz y video brindando a los usuarios una gran gama de posibilidades y exigiendo que estos sistemas tengan un desempeño eficiente. Es importante crear mecanismo para evaluar estos sistemas y así seguir haciéndolos mas eficientes. La simulación permite una solución práctica y económica para realizar dichas evaluaciones.

INTRODUCCIÓN

NECESIDAD DE LA EVALUACIÓN DE UN SISTEMA

Durante el ciclo de vida de un sistema, el cuál comprende el diseño, la adquisición, la explotación y la ampliación del mismo, es común que se lleven a cabo una o varias evaluaciones de las prestaciones que ofrece dicho sistema, durante las cuales hay que examinar que deficiencias y/o problemas se presentan en éste y solucionarlos sobre la marcha, o bien detectar que componentes del sistema es necesario cambiar para maximizar el aumento de prestaciones.

Dichas evaluaciones deben realizarse de forma objetiva, utilizando los parámetros relevantes del sistema, cuyos efectos puedan ser medidos y analizados a lo largo del tiempo o bien que puedan ser comparados con otros sistemas. Al realizar estas evaluaciones también es necesario contar con algunas expectativas sobre el uso del sistema, por ejemplo, cuantas conexiones simultáneas es capaz de soportar una base de datos, o el número de peticiones que puede manejar un sitio Web.

Los objetivos de una evaluación suelen ser los siguientes:

- Comparar alternativas.
- Determinar el impacto de una nueva característica.
- Sintonizar el sistema, es decir, hacer que funcione mejor según algún punto de vista.
- Identificar prestaciones relativas entre diferentes sistemas.
- Depuración de prestaciones, es decir, identificar los fallos del sistema que hacen que vaya más lento.

ORGANIZACIÓN DE LA TESIS

El presente trabajo de tesis tiene como finalidad evaluar el desempeño del protocolo Aloha con ayuda de la herramienta de simulación NS-2, estudiando los factores que intervienen en su desempeño en un escenario de red satelital.

En el capítulo 1 se analiza la metodología a utilizar para llevar a cabo dicha evaluación del desempeño del protocolo Aloha en un entorno de red satelital.

En el capítulo 2 se hace una descripción detallada del programa de simulación NS-2 y su utilidad para la evaluación de desempeños de diferentes entornos de red.

El capítulo 3 tratamos el tema de los protocolos de acceso múltiple en sistemas satelitales y también damos una descripción de cómo trabajan los algoritmos de acceso múltiple.

Las características del protocolo Aloha, los escenarios adecuados para la evaluación del desempeño y los resultados de las simulaciones se presentan en el capítulo 4.

ANTECEDENTES

La evaluación del desempeño se puede aplicar a cualquier cosa, para realizar primero se deben establecer los objetivos de la evaluación, qué herramientas vamos a utilizar para evaluar, y ver los resultados de la evaluación.

Cualquier sistema que sea diseñado debe satisfacer ciertas especificaciones de rendimiento asignadas previamente. Buscar las metodologías de diseño y procedimientos de la evaluación son necesarios para obtener sistemas que cumplan las especificaciones y obtener el mejor rendimiento con los menores costos. Se deben tomar en cuenta los factores del sistema, estos comprende los recursos, que son el conjunto de componentes de un sistema; los parámetros, que son el conjunto de atributos de cada recurso; las necesidades, que están en

función del trabajo que realizará el sistema; los costos, que es un factor indirecto pero que siempre hay que tener en cuenta; y por último el rendimiento.

Pasos en un estudio de evaluación del rendimiento:

- Establecer los objetivos del estudio y definir los límites del sistema.
- Enumerar los servicios del sistema y los resultados posibles.
- Seleccionar las métricas de rendimiento.
- Enumerar los parámetros del sistema y los parámetros de la carga de trabajo.
- Seleccionar los factores y sus valores. Los factores son los parámetros seleccionados en el paso anterior y que varían durante la evaluación.
- Los distintos valores se denominan niveles.
- Seleccionar las técnicas de evaluación.
- Seleccionar la carga de trabajo.
- Diseñar los experimentos.
- Determinar cuantos niveles tendrá cada factor.
- Cuantos experimentos se realizarán.
- Analizar e interpretar los datos.
- Presentar los resultados.

CAPÍTULO 1.

METODOLOGÍA

Inicialmente nos encontramos con la problemática de qué método o técnica de evaluación sería la más adecuada para lograr nuestro objetivo. La evaluación del desempeño es útil cuando se pretende optimizar el funcionamiento de los sistemas con el fin de proveer un servicio más eficiente de la red a los usuarios.

Para este propósito, existen tres técnicas a considerar:

- Modelado Analítico
- Simulación
- Medición

Las mediciones son ampliamente recomendadas cuando se cuenta con la infraestructura necesaria para efectuar la medición. Este método de evaluación es más preciso, es menos flexible y más costoso con respecto a los métodos por simulación y modelado analítico. El modelado analítico resulta conveniente cuando se pretende hacer abstracciones de fenómenos físicos, construyendo, inicialmente un modelo matemático que se aproxime al fenómeno de estudio en cuestión, por lo tanto estos sólo requieren papel, lápiz y tiempo del analista por lo que resulta ser el menos costoso. Por último, la simulación incorpora más detalles que permiten obtener una mejor descripción del objeto en estudio con relación al modelado analítico y es una buena solución cuando no se tiene la infraestructura necesaria para efectuar mediciones. La definición formulada por R.E. Shannon es:

> "*La simulación es el proceso de diseñar un modelo de un sistema real y llevar a término experiencias con él, con la finalidad de comprender el comportamiento del sistema o evaluar nuevas estrategias (dentro de los límites impuestos por un cierto criterio o un conjunto de ellos) para el funcionamiento del sistema*". [4]

A veces se utilizan simulaciones o mediciones para validar los resultados obtenidos a través de técnicas de modelado. La elección del método de análisis para este

trabajo se realizo por medio de la recopilación de artículos donde se describían los diferentes escenarios de red con sus respectivas características .Para validar los resultados del modelado analítico del protocolo Aloha vamos a utilizar la simulación, ya que no contamos con los recursos para llevar a cabo mediciones en el sistema real, además, éste método se acerca más a la realidad.

Los tipos de simulaciones conocidos son: Emulación, Simulación *Monte Carlo*, Simulación *trace-driven* y Simulación de eventos discretos.

Una simulación estática o sin variable en el tiempo es llamada simulación *Monte Carlo*. Este tipo de simulación se utiliza para fenómenos de modelo probabilístico que no cambian sus características con el tiempo. A diferencia de las simulaciones dinámicas que requieren la generación de números *seudo-aleatorios*. Las simulaciones *Monte Carlo* también son usadas para evaluar expresiones no-probabilísticas usando métodos probabilísticos.

Una simulación *trace-driven* se maneja generalmente por un archivo de traza. Éste tipo de simulación se utiliza usualmente en el análisis o valoración de los algoritmos de administración de recursos. Análisis de *cache*, algoritmos de planificación y algoritmos para dinámicas de asignación de recursos de la red son algunos ejemplos de casos donde las simulaciones *trace-driven* han sido usadas satisfactoriamente.

Una simulación utilizando un modelo de estado discreto del sistema es llamada simulación de eventos discretos; contrariamente a la simulación de eventos continuos, que es aquella en la que el estado del sistema toma valores constantes. Los modelos de estado continuo son usados en simulaciones químicas, donde el estado del sistema es descrito por la concentración de una sustancia química. En sistemas computacionales, se utilizan modelos de eventos discretos cuando el estado del sistema es descrito por el número de tareas en varios dispositivos.

Para tener una definición exacta del sistema que se desea simular, es necesario hacer primeramente un análisis preliminar de éste, con el fin de determinar la interacción con otros sistemas, las restricciones del sistema, las variables que interactúan dentro del sistema y sus interrelaciones, las medidas de efectividad que se van a utilizar para definir y estudiar el sistema y los resultados que se esperan obtener del estudio. [8]

Es importante que se definan con claridad y exactitud los datos que el modelo va a requerir para producir los resultados deseados.

Después debemos validar el modelo para detallar deficiencias en la formulación de éste o en los datos alimentados al modelo. Las formas más comunes de validar un modelo son:

- La opinión de expertos sobre los resultados de la simulación.
- La exactitud con que se predicen datos históricos.
- La exactitud en la predicción del futuro.

La experimentación con el modelo se realiza después que éste haya sido validado, se interpretan los resultados que arroja la simulación y con base a esto se toma una decisión.

Se requieren dos tipos de documentación para hacer un mejor uso del modelo de simulación. La primera se refiere a la documentación del tipo técnico y la segunda se refiere al manual del usuario, con el cual se facilita la interacción y el uso del modelo desarrollado.

En la Tabla 1-1 mostramos una breve descripción de algunos de los simuladores existentes.

El simulador elegido fue el *NS-2*, puesto que las cualidades que posee se ajustaban a nuestros requerimientos y presupuesto, es un simulador de eventos

discretos, de software libre, alta flexibilidad, auque no es muy amigable ni fácil de utilizar, nos permite realizar una gran variedad de aplicaciones.

Tabla 1-1 Simuladores

NOMBRE	DESCRIPCIÓN
MARS 2.1	Es escrito en C. Contiene dos interfaces, una simple y una gráfica. Carece de soporte estable. Simulación en tiempo discreto.
REAL	Minimiza el tiempo de ejecución comparado con versiones anteriores. Mejoras para analizar redes conmutadas. Limitado para análisis de sistemas de tiempo real.
J-Sim 1.3	Escrito en Java y Otcl. Permite la extensión y creación de nuevas clases en Java. Simulación mediante scripts. Permite el uso de casi todos los protocolos. Simulador en tiempo real y discreto.
Omnet. OMNet++	Escrito en C++. Interfaces de ejecución gráficas y de comandos. Es de fácil ampliación. Orientado a procesos y eventos. Simulador de eventos discretos.
S3 Project	(Scalable Simulation Framework) Escrito en C++ y Java. Tiene 2 interfaces escritas en estos lenguajes. Es escalable y permite el uso de casi todos los protocolos. Simulación discreta. No es libre y las versiones gratuitas son ineficientes.
Ns-2	Escrito en C++ y Otcl. Software libre. Simulador de eventos discretos. Expandible. Simulación mediante scripts. Permite la realización casi inmediata de simulaciones TCP/IP bastante completas. Esta avalado internacionalmente por diversas universidades además de ser una herramienta muy potente y se utiliza para probar los protocolos que serán implementados en un futuro.
Parsec	Lenguaje de programación basado en C, no tiene librerías y sólo se requiere del compilador. Parsec precisa de un código fuente, que compilado proporciona un ejecutable. Sin embargo, no existe tanta disponibilidad de modelos y librerías como en Ns u Omnet.

El NS-2 está avalado por las comunidades de investigación de Internet; quiénes utilizan esta herramienta desde hace varios años e incluso permite validar el funcionamiento de nuevos estándares de comunicaciones.

CAPÍTULO 2.

NETWORK SIMULATOR 2

El *Network Simulator 2* (*NS-2*) es un simulador orientado a objetos, escrito en C++, con un interpretador *OTcl* como interfase de usuario. El simulador soporta una jerarquía de clase en C++, (también llamada jerarquía compilada), y una jerarquía de clase similar dentro del interpretador *OTcl*, (llamada jerarquía interpretada). Las dos jerarquías están estrechamente relacionadas, desde las perspectiva del usuario, hay una correspondencia uno a uno entre una clase en la jerarquía interpretada y una en la jerarquía compilada. La raíz de ésta jerarquía es la clase *TclObject*. El usuario crea nuevos objetos de simulador a través del intérprete, estos objetos son creados dentro del intérprete, y fielmente reflejados por el objeto correspondiente en la jerarquía compilada.

2.1. CONCEPTO GENERAL

El NS-2 usa dos lenguajes debido a que el simulador tiene dos diferentes tipos de procesos que necesita hacer. Por una parte, la simulación detallada de protocolos requiere un sistema de lenguaje de programación que eficientemente manipule bytes, paquetes, cabeceras e implemente algoritmos que corran sobre largos grupos de datos. Para éstas tareas, la velocidad en los tiempos de ejecución es importante y los eventos de tiempo (iniciar simulación, encontrar y reparar errores, recopilación, y reinicio) son menos importantes.

Por otro lado, una gran parte de la investigación de redes involucra variaciones ligeras de parámetros o configuraciones, o exploración rápida de varios escenarios. En estos casos, los tiempos de iteración (cambios del modelo y reinicio) son más importantes.

El NS-2 reúne estas necesidades con dos lenguajes, C++ y *OTcl*. C++ es de ejecución rápida, pero lento para cambiar, lo que lo hace más adecuado para la implementación de los protocolos. *OTcl* es de ejecución mas lenta, pero puede cambiarse rápidamente, haciéndolo ideal para configurar de la simulación. El NS-2

genera la codificación para que los objetos y variables aparezcan en ambos lenguajes.

2.2. VISTA GENERAL DEL CÓDIGO

Existen varias clases definidas en *OTcl*, aquí mostraremos seis que son utilizadas en el NS-2:

La ***Clase Tcl*** encapsula las instancias reales de el interprete *OTcl*, y proporcionan los procesos para acceder y comunicarse con ése interprete. Los métodos de ésta clase son relevantes para el programador de NS-2 que este escribiendo códigos en C++. Estos métodos realizan las operaciones siguientes:

- Obtener una referencia de la instancia *Tcl*,
- Invocar procesos *OTcl* a través del interprete,
- Recuperar o regresar resultados al interprete,
- Reportar situaciones de error y salida de una manera uniforme,
- Guardar y consultar "*TclObjects*", y
- Adquirir acceso directo al intérprete.

La ***Clase TclObject*** es la clase base para varias de las otras clases en la jerarquía interpretada y en la jerarquía compilada. Cada objeto en esta clase se crea por el usuario dentro del intérprete. Un objeto reflejado equivalente es creado en la jerarquía compilada. Los dos objetos están estrechamente relacionados. La *Clase TclClass* contiene los mecanismos que generan este objeto reflejado.

Esta Clase se encarga de crear y destruir los objetos (también llamados *TclObjects*), de la unión de variables entre ambas jerarquías mediante los métodos de la clase *InstVar* y de la traza de variables.

Las siguientes secuencias de acciones son generadas por el intérprete como parte de la creación de un nuevo *TclObject*.

1. Obtener una identificación única para el nuevo objeto.
2. Ejecutar el constructor para el nuevo objeto. Cualquier argumento especificado por el usuario son pasados como argumentos al constructor.
3. El constructor del *TclObject* convoca los procesos para la creación del objeto reflejado.
4. Cuando el objeto reflejado se crea, el NS-2 llama a todos los constructores para dicho objeto, cada uno de los cuales puede establecer enlaces de variables para objetos en esa Clase, y generar otras inicializaciones necesarias.

La ***Clase TclClass*** es una clase compilada virtual pura. Clases derivadas de ésta clase base proveen dos funciones: construir la jerarquía de clase interpretada para reflejarla en la jerarquía de clase compilada; y proveer los procesos para crear nuevos *TclObjects*. La clase define solo el constructor, y un proceso adicional, el de crear elementos para el *TclObject* asociado.

La ***Clase TclCommand*** proporciona sólo el mecanismo para NS-2 de exportar comandos simples al intérprete, que pueden ejecutarse dentro de un contexto global por el interpretador. Ejemplos de este tipo de comandos pueden ser: *ns-version*, que entrega la versión del NS-2 que se este utilizando y*, ns-random* que entrega un valor al azar entre 0 y (2^{31} -1).

La ***Clase EmbeddedTcl*** permite la carga y evaluación de scripts, los cuales se generan para el desarrollo de funcionalidades en ambos códigos, tanto compilado como interpretado, eso se evalúa en la inicialización.

La ***Clase InstVar*** define los métodos y mecanismos para ligar una variable de C++ en el objeto reflejado compilado a una variable específica de *OTcl* en el objeto

interpretado equivalente. El enlace se monta de forma que el valor de la variable puede ser colocado o accesado desde el interpretador, o desde el código compilado en cualquier momento. [5]

La Figura 2-1 muestra la relación que existe entre las clases antes mencionadas dentro del simulador.

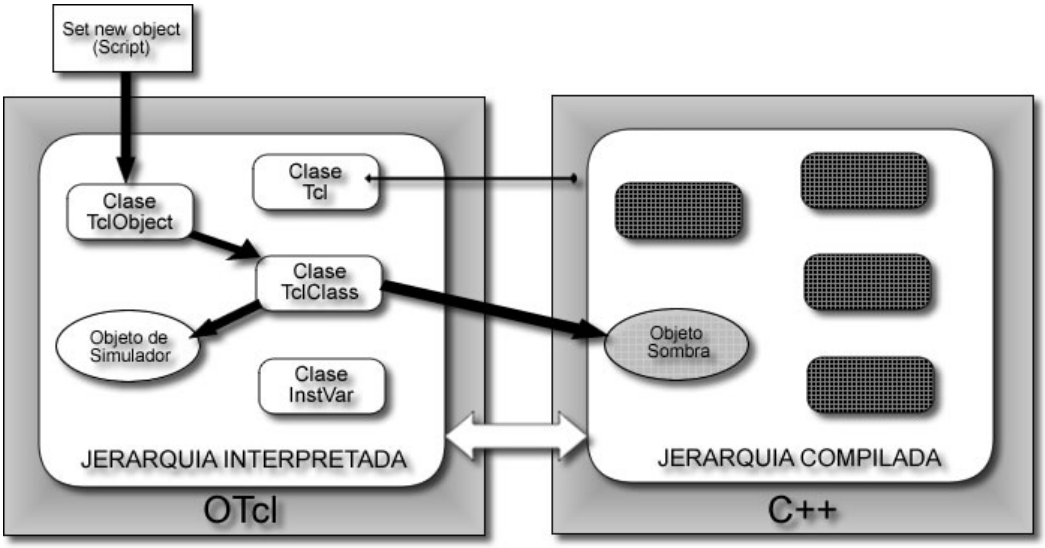


Figura 2-1 Relación de las Clases Principales en el NS-2

2.3. USO DE VARIABLES

En muchos casos, acceder a las variables del compilador esta restringido sólo al código compilado, y acceder a las variables del interprete están de igual forma confinadas a acceder sólo a través del código interpretado, sin embargo, es posible establecer enlaces bidireccionales de forma que las variables del interprete y las variables de compilador accedan el mismo dato, y que al cambiar el valor en cualquiera de ellos, cambie el valor de su pareja correspondiente al mismo valor. Estos enlaces se hacen a través de la *Clase InstVar*

El NS-2 soporta 5 diferentes tipos de datos: real, ancho de banda, tiempo, entero y boléanos. La sintaxis de cómo éstos valores pueden ser especificados en *OTcl* es diferente para cada tipo de variable y son explicados a continuación.

Variables Reales y enteros se especifican de la forma normal:

$objeto set valentero 10
$objeto set valreal 1.2e3

El ancho de banda se especifica como un valor real, opcionalmente pueden usarse los sufijos: *"k"* o "*K*" para indicar Kilo, "*m"* o "*M*" para indicar Mega, y "*B*" que indica que la cantidad esta expresada en Bytes. El valor preestablecido de ancho de banda se expresa en bits por segundo. Como ejemplo, las siguientes expresiones son equivalentes:

$objeto set bwvar 1.5M
$objeto set bwvar 1500K
$objeto set bwvar 1.5e6
$objeto set bwvar 0.1875mB
$objeto set bwvar 187.5kB

El tiempo se especifica como valor real, opcionalmente pueden usarse los sufijos: "*m*" para milisegundos, "*n*" para nanosegundos, o "*p*" para picosegundos. El valor predeterminado de tiempo se expresa en segundos.

$objeto set timevar 1.5

Los valores boléanos pueden expresarse de la siguiente forma: "*True*", "*t*" o "*1*", para indicar verdadero, y "*False*", "*junk*" o "*0*", para indicar falso. [6]

$objeto set boolvar true
$objeto set boolvar false

2.4. COMPONENTES DEL SIMULADOR

El NS-2 maneja un gran número de componentes que pueden ser utilizados y configurados al momento de generar la simulación, estos elementos van desde el tipo de nodo que se va a utilizar hasta el método de ruteo que se desea emplear, así como diferentes escenarios como los son las redes de área local, redes de área metropolitana, redes inalámbricas y redes satelitales.

A continuación se da una breve reseña de los componentes más comunes que se utilizan para crear un escenario de simulación.

2.4.1. NODOS

La forma básica para crear un nodo es mediante el comando:

set ns [new Simulator]
$ns ***node***

Esto crea la forma más simple del nodo y crea dos objetos internos, un clasificador de dirección, y un clasificador de puerto. La función de estos clasificadores es distribuir paquetes entrantes al agente correcto o al enlace de salida.

La configuración avanzada del nodo se realiza mediante el comando:

$ns node-config

Los componentes configurables a través de este comando permiten especificar un uso más concreto de los nodos, estos se muestran en la Tabla 2-1:

Tabla 2-1 Parámetros de configuración de los nodos

Opciones	Valores Avalibles	default
GENERAL		
addressType	flat, hierarchical	flat
MPLS	ON, OFF	OFF
BOTH SATELLITE- AND WIRELESS-ORIENTED		
wiredRouting	ON, OFF	OFF
llType	LL, LL/Sat	""
macType	Mac/802_11, Mac/Csma/Ca, Mac/Sat, Mac/Sat/UnslottedAloha, Mac/Tdma	""
ifqType	Queue/DropTail, Queue/DropTail/PriQueue	""
phyType	Phy/WirelessPhy, Phy/Sat	""
WIRELESS-ORIENTED		
adhocRouting	DIFFUSION/RATE, DIFFUSION/PROB, DSDV,DSR, FLOODING, OMNIMCAST, AODV, TORA	""
propType	Propagation/TwoRayGround, Propagation/Shadowing	""
propInstance	Propagation/TwoRayGround, Propagation/Shadowing	""
antType	Antenna/OmniAntenna	""
channel	Channel/WirelessChannel, Channel/Sat	""
topoInstance	<topology file>	""

Opciones	Valores Avalibles	default
mobileIP	ON, OFF	OFF
energyModel	EnergyModel	""
initialEnergy	<value in Joules>	""
rxPower	<value in W>	""
txPower	<value in W>	""
idlePower	<value in W>	""
agentTrace	ON, OFF	OFF
routerTrace	ON, OFF	OFF
macTrace	ON, OFF	OFF
movementTrace	ON, OFF	OFF
errProc	UniformErrorProc	""
FECProc	?	?
toraDebug	ON, OFF	OFF
SATELLITE-ORIENTED		
satNodeType	polar, geo, terminal, geo-repeater	""
downlinkBW	<bandwidth value, e.g. "2Mb">	""

2.4.2. ENLACES

Los enlaces se hacen mediante el siguiente comando:

$ns (link-type) (node0) (node1) (bandwidth) (delay) (queue_type)

A continuación se muestra una descripción de los elementos de la función anterior

- *(link-type)*, en este apartado se describe el tipo de enlace que se desea utilizar, los más comunes son *simplex-link* que crea un enlace unidireccional, y *duplex-link* que crea un enlace bidireccional.
- *(node0)(node1)*, representan los nodos entre los que se desea hacer el enlace.
- *(bandwidth)*, aquí se escribe el ancho de banda que tendrá dicho enlace.
- *(delay)*, en ésta parte se indica el tiempo que tardará cada paquete en recorrer este enlace.
- *(queue_type)*, aquí se maneja el tipo de algoritmo de encolamiento que va a utilizarse, los diferentes tipos de algoritmos existentes en NS-2 son *DROP-TAIL, FQ, SFQ, DRR, RED y CBQ.*

2.4.3. AGENTES

De igual forma el NS-2 permite el uso de varios protocolos de transporte, en la Tabla 2-2 se muestran estos así como los parámetros con que cuenta cada uno de ellos.

2.4.4. TRAFICO

Dentro del los tipos de tráfico que el NS-2 puede manejar se encuentran: tráfico exponencial, tráfico de Pareto, tráfico *CBR*, un tipo de tráfico que se genera a partir de un archivo externo.

La Tabla 2-3 muestra el tipo de tráfico, el comando que se utiliza y los parámetros que se configuran.

Tabla 2-2 Protocolos de Transporte

AGENTE	COMANDO	PARAMETROS (default values)	
UDP	set (nombre) [new Agent/UDP] $ns_ attach-agent <node> <nombre>	$(nombre) set packetSize_ 100	
TCP	set (nombre) [new Agent/TCP] $ns_ attach-agent <node> <agent>	$(nombre) set window_ 20 $(nombre) set windowInit_ 1 $(nombre) set windowOption_ 1) $(nombre) set windowConstant_ 4 $(nombre) set windowThresh_ 0.002 $(nombre) set overhead_ 0 $(nombre) set ecn_ 0 $(nombre) set packetSize_ 1000 $(nombre) set bugFix_ true $(nombre) set slow_start_restart_ true $(nombre) set tcpTick_ 0.1) $(nombre) set maxrto_ 64	$(nombre) set dupacks_ 0 $(nombre) set ack_ 0 $(nombre) set cwnd_ 0 $(nombre) set awnd_ 0 $(nombre) set ssthresh_ 0 $(nombre) set rtt_ 0 $(nombre) set srtt_ 0 $(nombre) set rttvar_ 0 $(nombre) set backoff_ 0 $(nombre) set maxseq_ 0
SCTP	set (nombre) [new Agent/SCTP] $ns attach-agent (nodo) (nombre)	$(nombre) set debugMask_ 0 $(nombre) set debugFileIndex_ -1 $(nombre) set associationMaxRetrans_ 10 $(nombre) set pathMaxRetrans_ 5 $(nombre) set changePrimaryThresh_ -1 $(nombre) set maxInitRetransmits_ 8 $(nombre) set oneHeartbeatTimer_ 1 $(nombre) set heartbeatInterval_ 30 $(nombre) set mtu_ 1500 $(nombre) set initialRwnd_ 65536 $(nombre) set initialSsthresh_ 65536 $(nombre) set initialCwnd_ 2 $(nombre) set initialRto_ 3.0 $(nombre) set minRto_ 1.0 $(nombre) set maxRto_ 60.0	$(nombre) set fastRtxTrigger_ 4 $(nombre) set numOutStreams_ 1 $(nombre) set numUnrelStreams_ 0 $(nombre) set reliability_ $(nombre) set unordered_ 0 $(nombre) set ipHeaderSize_ 20 $(nombre) set dataChunkSize_ 1468 $(nombre) set useDelayedSacks_ 1 $(nombre) set sackDelay_ 0.200 $(nombre) set useMaxBurst_ 1 $(nombre) set rtxToAlt_ 1 $(nombre) set dormantAction_ 0 $(nombre) set routeCalcDelay_ 0 $(nombre) set routeCacheLifetime_ 1.2 $(nombre) set trace_all_ 0
SRM	set (nombre) [new Agent/SRM] $ns attach-agent (nodo) (nombre) set grp [Node allocaddr] $srm set dst_ $grp	$(nombre) set C1_ 2.0 $(nombre) set C2_ 2.0 $(nombre) set D1_ 1.0 $(nombre) set D2_ 1.0 $(nombre) set requestFunction_ "SRM/request"	$(nombre) set repairFunction_ "SRM/repair" $(nombre) set sessionFunction_ "SRM/session" $(nombre) set requestBackoffLimit_ 5 $(nombre) set sessionDelay_ 1.0
PLM	set srm [new $(nombre)] $ns attach-agent (nodo) (nombre) set grp [Node allocaddr] $srm set dst_ $grp	set packetSize 500 set plm_debug_flag 2 set rates "50e3 50e3 50e3 50e3 50e3 set rates_cum [calc_cum $rates set level [llength $rates set Queue_sched_ FQ set PP_burst_length 2 set PP_estimation_length 3 Class Scenario0 -superclass PLMTopology Scenario0 instproc init args {	eval $self next $args $self instvar ns node $self build_link 1 2 100ms 256Kb set addr(1) [$self place_source 1 3] $self place_receiver 2 $addr(1) 5 DM set PruneTimeout set mproto DM set mrthandle [$ns mrtproto $mproto {}]

Tabla 2-3 Tipos de Tráfico

TRAFICO	COMANDO	PARAMETROS
EXPONENCIAL	set (nombre) [new Application/Traffic/Exponential]	$(nombre) set packetSize_ 210 $(nombre) set burst_time_ 500ms $(nombre) set idle_time_ 500ms $(nombre) set rate_ 100k
PARETO	set (nombre) [new Application/Traffic/Pareto]	$(nombre) set packetSize_ 210 $(nombre) set burst_time_ 500ms $(nombre) set idle_time_ 500ms $(nombre) set rate_ 200k $(nombre) set shape_ 1.5
CBR	set (nombre) [new Application/Traffic/CBR]	$(nombre) set packetSize_ 48 $(nombre) set rate_ 64Kb $(nombre) set random_ 1 $(nombre) set interval_ 0.1 $(nombre) maxpkts_ 1000
TRACEFILE	set (nombre) [new Tracefile] $(nombre) filename example-trace set (nombre1) [new Application/Traffic/Trace] $(nombre1) attach-tracefile $(nombre) set (nombre2) [new Application/Traffic/Trace] $(nombre2) attach-tracefile $(nombre)	

2.5. ARCHIVOS DE TRAZA

El NS-2 cuenta con varias formas de entregar resultados; cada evento generado en la simulación se guarda en un archivo el cuál se conoce como archivo de traza. Dentro de los tipos existentes, los que principalmente se utilizan son: el archivo de traza básico y el archivo de traza *NAM*.

2.5.1. ARCHIVO DE TRAZA BÁSICO

Este archivo contiene el registro de los eventos generados durante la simulación. La Figura 2-2 muestra la forma en que se presentan los datos, se muestran 14 eventos, la primera columna muestra el tipo de evento donde: '+' indica entrada en *buffer*, '-' salida de *buffer*, 'r' indica paquete recibido,'c' muestra una colisión, y 'd'

que revela un paquete arrojado. La siguiente columna muestra el tiempo (en segundos) en el que sucedió dicho evento. Los siguientes dos campos indican los dos nodos entre los que ocurre el evento. La columna que sigue muestra el tipo de paquete y es seguido por el tamaño del mismo.

El siguiente campo contiene las banderas, que en este ejemplo no se usan. A continuación se muestra el identificador de flujo IP como esta definido para la versión 6 de IP. Los dos campos subsecuentes indican la fuente del paquete y el nodo destino respectivamente. El campo que le sigue indica el número de secuencia de paquete entre esos dos nodos. El último campo es un identificador único, a cada paquete creado se le asigna un identificador único.

+	1.84375	0	2	cbr	210	-------	0	0.0	3.1	225	610
-	1.84375	0	2	cbr	210	-------	0	0.0	3.1	225	610
r	1.84471	2	1	cbr	210	-------	1	3.0	1.0	195	600
r	1.84566	2	0	ack	40	-------	2	3.2	0.1	82	602
+	1.84566	0	2	tcp	1000	-------	2	0.1	3.2	102	611
-	1.84566	0	2	tcp	1000	-------	2	0.1	3.2	102	611
r	1.84609	0	2	cbr	210	-------	0	0.0	3.1	225	610
+	1.84609	2	3	cbr	210	-------	0	0.0	3.1	225	610
d	1.84609	2	3	cbr	210	-------	0	0.0	3.1	225	610
-	1.8461	2	3	cbr	210	-------	0	0.0	3.1	192	511
r	1.84612	3	2	cbr	210	-------	1	3.0	1.0	196	603
+	1.84612	2	1	cbr	210	-------	1	3.0	1.0	196	603
-	1.84612	2	1	cbr	210	-------	1	3.0	1.0	196	603
+	1.84625	3	2	cbr	210	-------	1	3.0	1.0	199	612

Figura 2-2 Formato del archivo de Traza Básico

Este archivo de traza se genera mediante el siguiente comando

$ns trace-all (nombre_del_archivo)

Una variante de este archivo de traza se utiliza para las redes de tipo satelital. A este archivo se le agregan 4 campos al final de cada línea, los dos primeros indican la posición (latitud y longitud respectivamente) del punto subsatélite del nodo

fuente, y los siguientes dos campos las posición del punto subsatélite del nodo destino.

+	1.0000	66	26	cbr	210	-------	0	66.0	67.0	0	0	37.90	-122.30	48.90	-120.94

Este tipo de archivo se genera mediante el comando

$ns trace-all-satlinks (nombre_del_archivo)

2.5.2. ARCHIVO DE TRAZA *NAM*.

Al igual que el archivo de traza común, el archivo de traza *NAM* contiene el registro de cada evento ocurrido durante la simulación, pero se genera en un formato que puede ser leído por el programa de animación de redes (*NAM*) que se incluye en el NS-2. Un ejemplo del formato generado se muestra en Figura 2-3:

r	-t	0.255	-s	1	-d	-1	-p	MAC	-e	512	-c	0	-a	0	-i	0	-k	MAC
+	-t	0.255	-s	1	-d	0	-p	AODV	-e	512	-c	0	-a	0	-i	0	-k	MAC
-	-t	0.255	-s	1	-d	0	-p	AODV	-e	512	-c	0	-a	0	-i	0	-k	MAC
h	-t	0.255	-s	1	-d	0	-p	AODV	-e	512	-c	0	-a	0	-i	0	-k	MAC
r	-t	0.255	-s	0	-d	1	-p	AODV	-e	512	-c	0	-a	0	-i	0	-k	MAC

Figura 2-3 Formato de los archivos de traza NAM.

Lamentablemente explicar todas las variables que se encuentran en este tipo de formato haría esta sección demasiado extensa, se recomienda al lector revisar el manual [5] para obtener información detallada de este tipo de formato.

Para obtener este tipo de archivos se utiliza el siguiente comando:

$ns namtrace-all (nombre_del_archivo)

Existe una variante existe para las redes inalámbricas que se realiza mediante el comando

$ns namtrace-all-wireless (nombre_del_archivo) (x) (y)

donde ‘x’ ‘y’, indican el tamaño en coordenadas del escenario de simulación. [5]

2.6. CREACIÓN DE UN ESCENARIO DE SIMULACIÓN

En esta sección mostraremos el desarrollo de un *Script TCL* para NS-2, explicando cada una de las funciones de los comandos básicos en la creación de un escenario de simulación.

Para iniciar, es necesario escribir el siguiente comando, el cual es necesario para la creación de un nuevo objeto de simulador.

```
set ns [new Simulator]
```

Los siguientes comandos sirven para la creación de un archivo de traza que podrá utilizarse por el *NAM* (Network Animator) donde se van a guardar los resultados generados. La primera línea se encarga de crear el archivo de salida *‘out.nam’*, mientras que la segunda línea indica al simulador que los datos deberán guardarse en el formato reconocido por el *NAM*.

```
set nf [open out.nam w]
$ns namtrace-all $nf
```

El siguiente paso es crear el proceso *‘finish’* que cerrara el archivo de traza e iniciara la animación con *NAM*.

```
proc finish {} {
        global ns nf
        $ns flush-trace
        close $nf
        exec nam out.nam &
exit 0
}
```

Los siguientes comando se encargan de crear los nodos que se utilizaran en el simulador, para este caso los nombres que se les asignan a estos nodos son *n0* y *n1*.

```
set n0 [$ns node]
set n1 [$ns node]
```

Lo siguiente es crear la conexión ente los dos nodos, esto se hace con el siguiente comando.

```
$ns duplex-link $n0 $n1 1Mb 10ms DropTail
```

Esta línea le dice al simulador que conecte los nodos *n0* y *n1*, con un enlace duplex, utilizando 1Mb de ancho de banda, un retardo de línea de 10ms y un algoritmo de encolamiento de tipo *DropTail*.

A continuación se va a crear un protocolo (agente) que va a enviar datos desde el nodo *n0* mediante los siguientes comandos. La primer línea crea el agente con el nombre *udp0,* mientras que la segunda línea relaciona ese agente al nodo *n0.*

```
set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0
```

La siguiente secuencia de comandos agrega un generador de tráfico *CBR* al agente UDP.

```
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
$cbr0 set interval_ 0.005
$cbr0 attach-agent $udp0
```

La primera línea crea el tipo de tráfico *CBR* con el nombre *cbr0*. Las siguientes dos líneas indican que el tamaño de los paquetes van a ser de 500 bytes, y un paquete va a ser enviado cada 0.005 segundos. La última línea relaciona el tráfico *CBR* con el agente *udp0*

Los dos comandos siguientes crean el sumidero (o receptor) con el nombre *null0* y se lo asigna al nodo *n1*.

```
set null0 [new Agent/Null]
$ns attach-agent $n1 $null0
```

Ahora es necesario introducir el comando que conecte el agente *udp0*, con el agente *null0*, es decir con el sumidero.

```
$ns connect $udp0 $null0
```

Los comandos siguientes indican en que momento inicia y termina el envío de datos del agente *cbr0*.

```
$ns at 0.5 "$cbr0 start"
$ns at 4.5 "$cbr0 stop"
```

La siguiente línea indica al simulador que ejecute el proceso *'finish'* después de 5 segundos de iniciada la simulación.

```
$ns at 5.0 "finish"
```

La última línea comienza la simulación.

```
$ns run
```

Para correr este programa debe entrar en la consola de comandos, dirigiéndose a la carpeta donde se haya guardado el archivo y teclear el siguiente comando.

```
ns nombre.tcl
```

Esto creara el archivo *out.nam*, y mostrará en pantalla la simulación mediante *NAM*, que mostrara una pantalla similar a la Figura 2-4. [6]

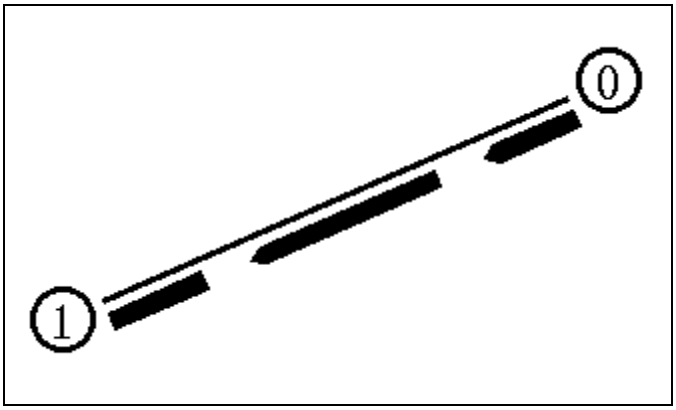


Figura 2-4 Pantalla de NAM

CAPÍTULO 3.
PROTOCOLO DE ACCESO MÚLTIPLE

Una de las ventajas de los sistemas de comunicaciones satelitales sobre otros medios de transmisión es su habilidad de conectar a un gran número de estaciones terrenas, proveyendo una comunicación punto-multipunto. El transpondedor de un satélite puede ser accesado por muchas estaciones terrenas, y por lo tanto es necesario tener técnicas para permitir el acceso a la capacidad del transpondedor para cada una de ellas. Además, para evitar el caos, quisiéramos que las estaciones terrenas ganaran acceso a la capacidad del transpondedor asignada a ellos en una manera ordenada. A estas técnicas se les llama Acceso Múltiple [1]. Los dos esquemas de acceso múltiple mas utilizados son los de Acceso Múltiple por División de Frecuencia (*FDMA*) y Acceso Múltiple por División de Tiempo (*TDMA*) y serán revisados posteriormente en éste capítulo.

3.1. ARQUITECTURA

Un protocolo de acceso múltiple o algoritmo de acceso múltiple (*MAA*) son las reglas por las cuales el usuario conoce cómo usar el tiempo, la frecuencia y funciones de código para comunicarse, a través de un satélite u otro recurso único, con otros usuarios. Un sistema de acceso múltiple es la combinación de hardware y software que soporta algoritmo de acceso múltiple. La finalidad del acceso múltiple es proveer un servicio de comunicación ordenada en tiempo y de forma eficiente.

Las formas de elegir la arquitectura de un sistema de acceso múltiple satelital se muestran a continuación con sus respectivas figuras:

En la Figura 3-1 una estación terrena es designada como maestra o controladora. Esta estación responde el pedido de servicio de todos los demás usuarios. Un pedido de servicio de usuario conlleva a la transmisión por satélite y devuelta al controlador. La respuesta del controlador implica otra transmisión satelital.

Entonces hay 2 transmisiones *up y downlink* requeridas para cada asignación de servicio.

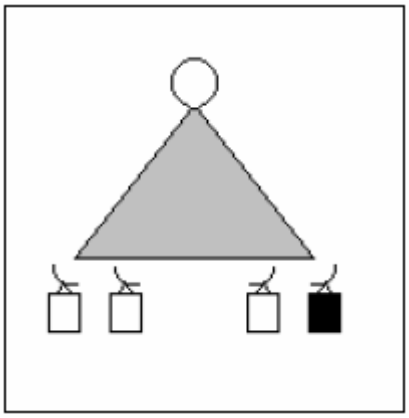


Figura 3-1 Estación terrena funcionando como controladora maestra

En la Figura 3-2 no hay un único controlador sino que está distribuido entre todas las estaciones. Cada estación terrena usa el mismo algoritmo y todas tienen el mismo conocimiento con respecto al pedido de acceso y asignaciones. Por lo tanto, sólo una única vuelta (*up y downlink*) es necesaria para una asignación.

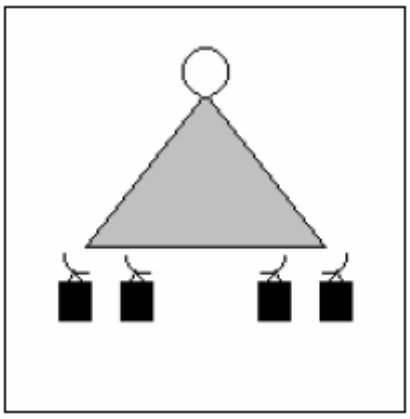


Figura 3-2 Algoritmo distribuido en todas las estaciones terrenas

En el caso de la Figura 3-3, el pedido de servicio va desde el usuario al satélite y la respuesta del satélite puede llegar inmediatamente. También es necesaria una única vuelta (*up y downlink*) para una asignación. [7]

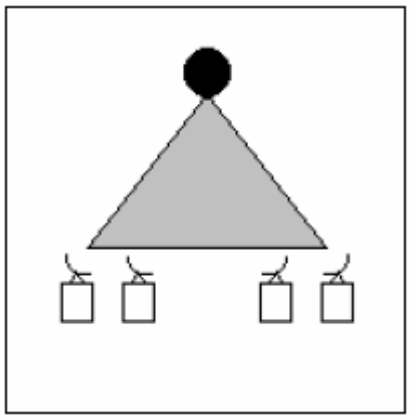


Figura 3-3 Satélite funcionando como controlador maestro

3.2. FLUJO DE INFORMACIÓN EN UN ACCESO MÚLTIPLE

En la Figura 3-4 se muestra un diagrama de flujo que describe el flujo básico de información entre el algoritmo de acceso múltiple y una estación terrestre. Los siguientes números corresponden a los de la Figura 3-4.

1) *Canalización:* se refiere a la distribución de información. Ejemplo: canales de 1 a N pueden estar asignados a la Armada, y los canales (N+1) a M a la Marina. Esta información raramente cambia y cuando lo hace, es distribuida a todas las estaciones terrenas por carta y no por el sistema de comunicación.

2) *Estado de la red:* Cada estación es notificada de la disponibilidad del recurso de comunicación y en qué forma está disponible (*FDMA, TDMA, CDMA*).

3) *Pedido de servicio:* luego la estación hace un pedido de servicio (reserva de N time slots).

4) Cuando se recibe el pedido, el controlador manda a la estación un cronograma informando cuando y dónde introducir su mensaje.

5) La estación transmite su mensaje.

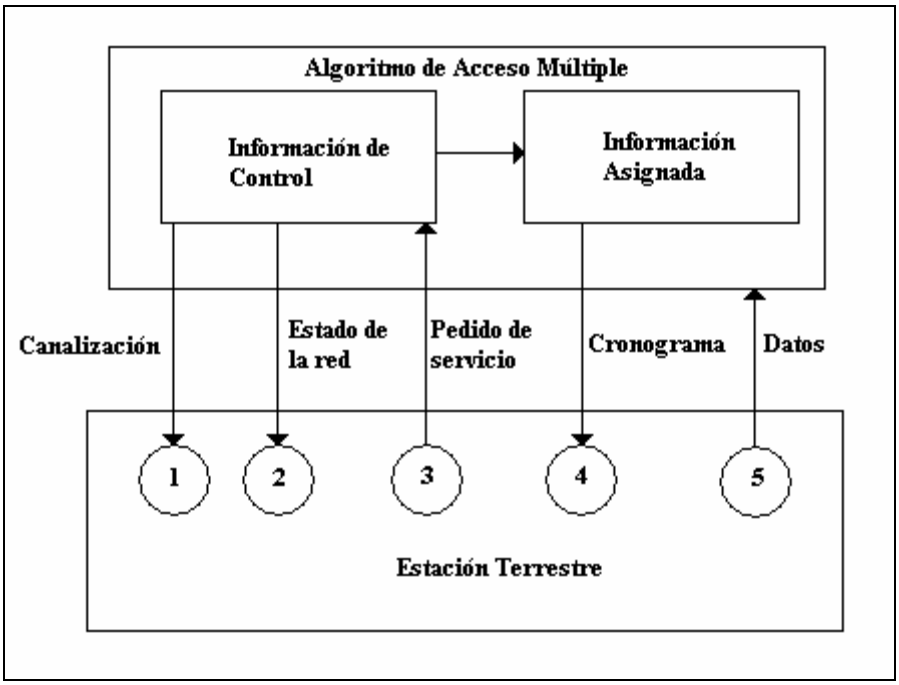


Figura 3-4 Diagrama de flujo de un algoritmo de acceso múltiple

3.3. ACCESO MÚLTIPLE POR DIVISIÓN DE FRECUENCIA (FDMA).

El acceso múltiple por división de frecuencia es un método de acceso múltiple en el que determinado ancho de banda de RF se divide en bandas menores de frecuencia, llamadas subdivisiones. Cada subdivisión tiene su propia frecuencia de portadora. Se usa un mecanismo de control para asegurar que dos o más estaciones terrenas no transmitan en la misma subdivisión al mismo tiempo. En esencia, el mecanismo de control designa una estación receptora para cada una de las subdivisiones. En sistemas de asignación por demanda, el mecanismo de control también se usa para establecer o terminar los enlaces de banda de voz entre estaciones terrenas de origen y destino. En consecuencia, cualquiera de las estaciones terrenas participantes pueden usar cualquiera de las subdivisiones en cualquier momento. Si cada subdivisión sólo porta un canal de banda de voz de 4

kHz, a esto se le llama sistema de un canal por portadora (*SCPC*, *single-channel per carrier*). Cuando se multiplexan varios canales de banda de voz por división de frecuencia para formar una señal compuesta de banda base formada por grupos, supergrupos o hasta grupos maestros, se asigna una subdivisión más ancha. A esto se le llama múltiples canales por portadora (*MCPC*, *multiple-channel per carrier*).

Las frecuencias y anchos de banda para los sistemas satelitales *FDM/FM* que usan formatos de varios canales por portadora se suelen asignar y permanecen fijos durante un largo tiempo. A esto se le llama asignación fija, acceso múltiple (*FDM/FM/FAMA*, *fixed-assignment, multiple access*).

Los tipos de modulación usados en *FDMA* son *FM* o *PSK*. La Figura 3-5 muestra un esquema de *FDMA.* [10]

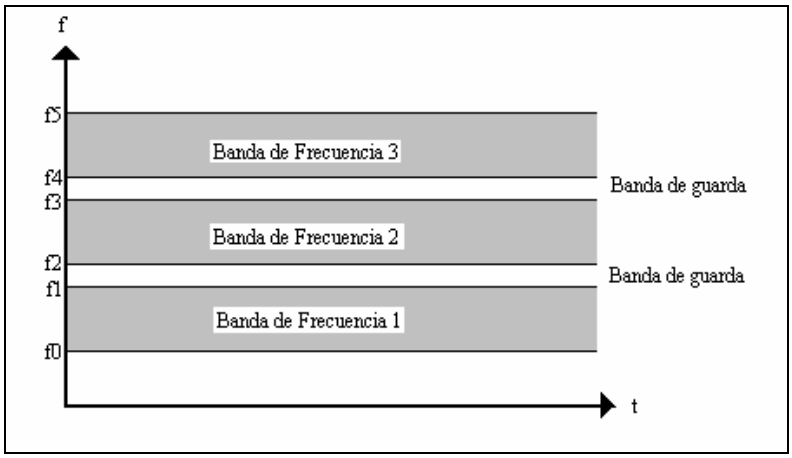


Figura 3-5 Esquema de la Técnica de Acceso Múltiple FDMA

3.4. ACCESO MÚLTIPLE POR DIVISIÓN DE TIEMPO (TDMA).

El acceso múltiple por división de tiempo es el método principal de acceso múltiple que se usa en la actualidad. Proporciona la forma más eficiente de transmitir portadoras moduladas digitalmente (*PSK*). El *TDMA* es un método de multiplexado por división de tiempo que multiplexa portadoras moduladas digitalmente entre las

estaciones terrenas participantes en una red satelital, a través de un transpondedor satelital común. En *TDMA*, cada estación terrena transmite una ráfaga corta de una portadora modulada digitalmente, durante una ranura precisa de tiempo dentro de una trama *TDMA*. La ráfaga de cada estación se sincroniza de tal modo que llegue al transpondedor del satélite en distinto momento. En consecuencia, solo hay una portadora de estación terrena presente en el transpondedor en cualquier momento, y se evita así una colisión con la portadora de otra estación. El transpondedor es una repetidora de RF a RF que sólo recibe las transmisiones de la estación terrena, las amplifica y a continuación las retransmite en un haz de enlace de bajada, que reciben todas las estaciones participantes. Cada estación terrena recibe ráfagas de todas las demás estaciones, y debe seleccionar entre ellas el tráfico destinado a ella. La Figura 3-6 muestra una trama *TDMA*.

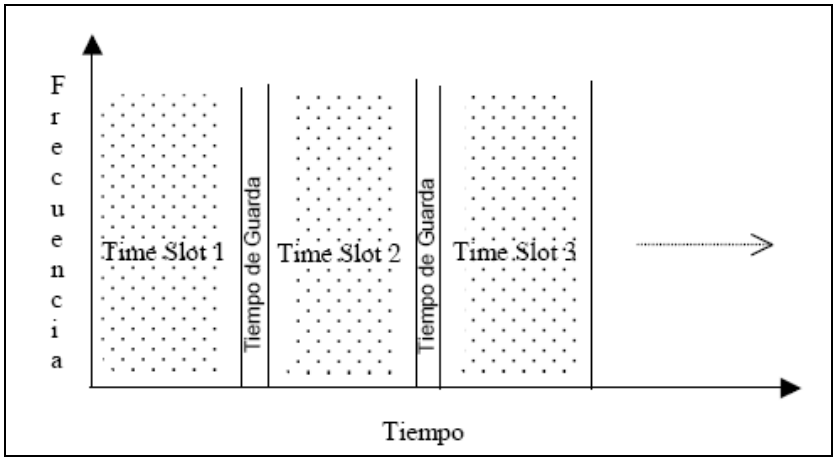


Figura 3-6 Esquema de la Técnica de Acceso Múltiple TDMA

Las transmisiones de todas las estaciones terrenas se sincronizan a una ráfaga de referencia. La ráfaga de referencia puede estar como transmisión separada, pero puede ser el preámbulo que antecede la transmisión de datos de una estación de referencia. También puede haber más de una ráfaga de referencia para sincronización.

La ráfaga de referencia contiene una secuencia de recuperación de portadora (*CRS, carrier recovery sequence*), de la cual todas las estaciones receptoras recuperan una portadora de frecuencia y fase coherentes para su desmodulación por *PSK*. También se incluye en la ráfaga de referencia una secuencia binaria para recuperación de sincronización de bits (*BTR, bit timing recovery*), es decir, para recuperación de reloj. Al final de cada ráfaga de referencia se transmite una palabra única (*UW, unique word*). La secuencia *UW* se usa para establecer una referencia precisa de tiempo que usa cada una de las estaciones terrenas para sincronizar la transmisión de su ráfaga. La *UW* suele ser una cadena de unos 20 binarios sucesivos, terminada con un 0 binario. Cada receptor de estación demodula e integra la secuencia *UW*.

Cada estación antecede con un preámbulo a la transmisión de sus datos. El preámbulo es lógicamente equivalente a la ráfaga de referencia. Como se deben recibir las transmisiones de cada estación por las demás estaciones, todas las estaciones deben recuperar la información de la portadora y del reloj, antes de demodular los datos. Si se usa asignación por demanda, también se debe incluir un canal común de señalización de preámbulo. [10]

3.5. ACCESO MÚLTIPLE POR DIVISIÓN ESPACIAL Y DE POLARIZACIÓN (SDMA Y PDMA)

En la Figura 3-7, en la parte ‘a’ se puede apreciar la técnica *SDMA* (también llamado *múltiple beam frequency reuse*). Este concepto implica la reutilización de frecuencias con aislamiento espacial. El satélite transmite con el mismo bloque de frecuencias F1 en las regiones 1 y 3.

En la parte ‘b’ se aprecia la técnica de PDMA (también llamado *dual polarization frequency reuse*). El satélite posee dos antenas, cada una con distinta polarización y receptores separados, lo que permite acceso simultáneo del satélite de la misma región de la Tierra. Esto obliga a que cada una de las estaciones terrenas tenga

sus antenas con la misma polarización que la del receptor del satélite que corresponda. Como en *SDMA*, la banda de frecuencias puede reutilizarse.

También es posible utilizar una técnica simultánea de *SDMA* y *PDMA*. Hay cubrimiento de dos hemisferios separados. También hay dos zonas más chicas, que se superponen con una porción de uno de los hemisferios y es separado de está por polarización ortogonal. En éste caso hay un cuádruple solapamiento del espectro. [11]

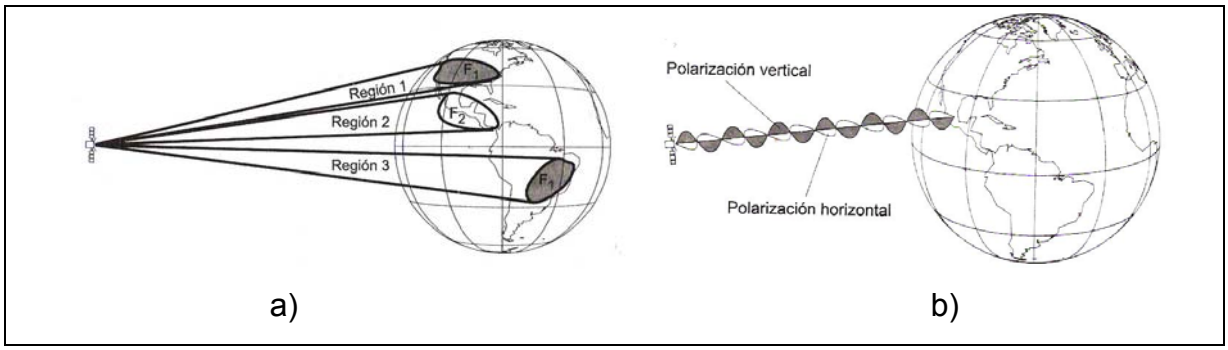


Figura 3-7 a) Técnica SDMA; b) Técnica PDMA

3.6. ACCESO MULTIPLE POR ASIGNACION DE DEMANDA (DAMA)

Un procedimiento de acceso múltiple con asignación fija es cuando una estación tiene acceso periódico al canal independientemente de su necesidad actual. Sin embargo, un procedimiento de acceso múltiple con asignación dinámica o también llamado *DAMA*, es cuando se le da a la estación acceso al canal sólo cuando se demanda el acceso. Si el tráfico es intermitente o tipo ráfaga, el procedimiento *DAMA* puede ser mucho más eficiente que una asignación fija. Si la demanda pico del sistema iguala a la capacidad del sistema y si el tráfico es del tipo ráfaga, el sistema está la mayor parte del tiempo sin ser aprovechado totalmente. Sin embargo, por el uso de *buffers* y el procedimiento *DAMA*, un sistema con capacidad reducida puede manejar un tráfico del tipo ráfaga, a costa de retardos debidos al *buffer*. De esta forma, se utiliza un canal con una capacidad igual al

promedio de los requerimientos de los usuarios. Por otro lado, en asignación fija, la capacidad es igual a la suma de los requerimientos máximos de cada usuario. [11]

3.7. ACCESO MULTIPLE POR DIVISION DE CODIGO (CDMA)

Aunque las técnicas de acceso múltiple *FDMA* y *TDMA*, son las de mayor uso en los satélites comerciales de comunicaciones de servicio fijo, con la técnica *CDMA* se puede compartir un transpondedor completo o parte de él, al ser ocupado por varias estaciones que transmiten a la misma frecuencia y al mismo tiempo. *CDMA* es particularmente útil en transmisiones confidenciales o altamente sensitivas a la interferencia; es usada en satélites militares y en satélites comerciales de servicio móvil. Aunque la utilización del espectro es baja, ya que las señales ocupan simultáneamente un ancho de banda muy grande, se logra obtener un "blindaje" contra el deterioro que pudiera ocurrir sobre las portadoras moduladas, causado por interferencias debidas a trayectorias múltiples, o por la misma interferencia propia del medio donde los sistemas móviles operan.

Esta también es una técnica digital como *TDMA*, y presenta la ventaja de que las antenas transmisoras y receptoras pueden ser muy pequeñas, sin importar que sus ganancias sean bajas y sus haces de radiación muy amplios.

En un sistema *CDMA* cada estación transmisora utiliza una secuencia diferente de bits para codificar cada uno de los bits de información; como cada señal ocupa un ancho de banda muy grande, su densidad espectral de potencia es muy baja.

De las estaciones terrenas receptoras, sólo la destinataria de cierta información determinada conoce el código con el que fue transmitida y es capaz de reconstruir el mensaje original, aunque llegue superpuesto con todos los demás mensajes que fueron transmitidos simultáneamente, pues estos últimos solo los detecta como "ruido" térmico, aleatorio y tolerable. [11]

CAPÍTULO 4.

PROTOCOLO ALOHA

En 1971 la universidad de Hawai comenzó la operación de su sistema Aloha. Se usó un satélite de comunicaciones para conectar distintas computadoras de la universidad usando un protocolo de acceso aleatorio. El concepto del sistema era bastante sencillo y consistía de los siguientes modos:

1. *Modo de transmisión.* El usuario transmite en el momento que desee, usando un código de detección de error.

2. *Modo escucha.* Después de la transmisión de un mensaje, un usuario espera (escucha) la transmisión de un acuse de recibo (*ACK*). Las transmisiones desde diferentes usuarios pueden solaparse en el tiempo causando errores de recepción de los datos. En tal caso se dice que el mensaje ha *colisionado*. Cuando esto ocurre, se detecta el error, y el usuario recibe un acuse de recibo negativo (*NAK*).

3. *Modo retransmisión.* Cuando se recibe un *NAK* entonces el mensaje es retransmitido. Se deja pasar un cierto tiempo aleatorio antes de la retransmisión, ya que si el usuario victima de la colisión transmitiera inmediatamente entonces podría sufrir otra colisión.

4. *Modo timeout.* Si, después de una transmisión, el usuario no recibe ni *ACK* ni *NAK* dentro de un determinado tiempo, entonces se vuelve a transmitir el mensaje.

La característica fundamental de Aloha es que no espera el “turno” para transmitir, como en el caso de *TDMA* ni tampoco usa una fracción del ancho de banda como en *FDMA*. En Aloha cada usuario transmite cuando quiere y usa todo el ancho de banda disponible. [1][7]

4.1. ESTADÍSTICAS DE ARRIBO DE MENSAJES

Todo el sistema demanda en promedio, λ mensajes exitosos por segundo. Dado la existencia de colisiones, algunos mensajes serán erróneos. La cantidad de mensajes erróneos (todos los mensajes presentes en la colisión) por segundo es λ_r. Se define λ_t como la tasa de tráfico de arribo total.

$$\lambda_t = \lambda + \lambda_r$$

Supongamos que el largo de cada mensaje o paquete es de *b* bits.

Se define ρ' como la cantidad promedio de bits exitosos que arriban al receptor (tasa de tránsito o *throughput*):

$$\rho' = \lambda \times b$$

G' como el tráfico total:

$$G' = \lambda_t \times b$$

ρ como la tasa de tránsito normalizada

$$\rho = \frac{\lambda \times b}{C}$$

$$0 \leq \rho \leq 1$$

C: capacidad del canal [*bits/s*] e interpretada como la tasa binaria máxima que puede manejar el sistema.

G como tráfico total normalizado

$$G = \frac{\lambda \times b}{C}$$

$$0 \leq G \leq \infty$$

A diferencia de ρ, G puede ser mayor que uno.

τ como tiempo de transmisión de cada palabra o paquete:

$$\tau = \frac{b}{C}$$

Reemplazando la ecuación de definición de τ en la de ρ y G queda:

$$\rho = \lambda \times \tau$$

$$G = \lambda_t \times \tau$$

Es posible transmitir mensajes exitosos mientras ningún otro mensaje intente ocupar el recurso de comunicación en τ segundos antes y τ segundos después del inicio de la transmisión. Esto se puede observar en la Figura 4-1. [1][2]

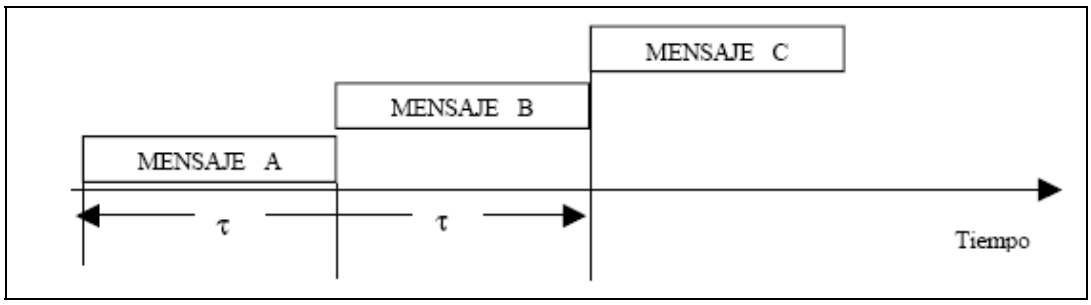


Figura 4-1 Transmisión de paquetes sin colisiones

Pero en el algoritmo cada mensaje necesita de 2 τ segundos de espacio libre como mínimo, de lo contrario existirá una colisión. Esto se puede observar en la Figura 4-2.

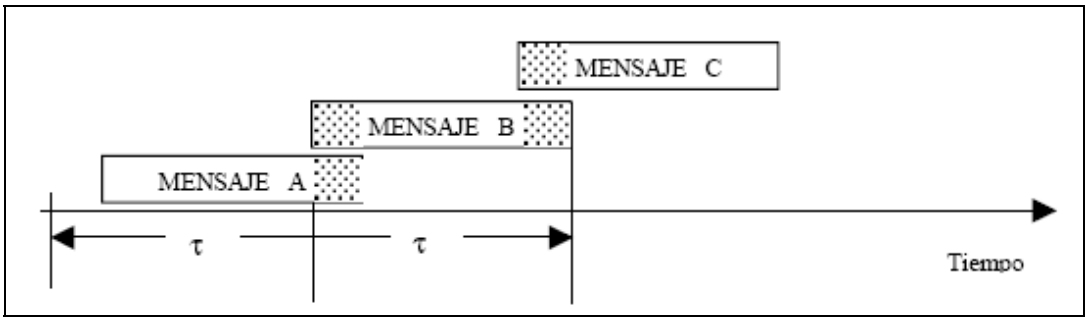


Figura 4-2 Colisión de paquetes

La recepción de mensajes, mientras la transmisión de cada uno sea independiente de los otros, se puede modelar mediante un proceso de *Poisson*. La probabilidad de recibir *K* mensajes nuevos en *t* segundos, está dado por la distribución de *Poisson*:

$$P(K) = \frac{(\lambda \cdot t)^K e^{-\lambda \cdot t}}{K!}$$

$$K > 0$$

Es interesante calcular P(K=0)=*Ps* con t=2τ, para conocer la probabilidad de recibir un mensaje exitoso (sin colisión).

$$P_s = \frac{(\lambda_t 2\tau)^0 e^{-\lambda_t 2\tau}}{0!} = e^{-2\tau\lambda_t}$$

Por definición, $P_s = \lambda / \lambda_t$

Igualando las últimas dos ecuaciones, se obtiene:

$$\lambda = \lambda_t e^{-2\tau\lambda_t}$$

Recordando que $\rho = \lambda\tau$ y que $G = \lambda_t \tau$, se llega a:

$$\rho = Ge^{-2G}$$

Esta relación recibe el nombre de "*Pure Aloha*". La Figura 4-3 muestra la gráfica de eficiencia de este sistema. [7]

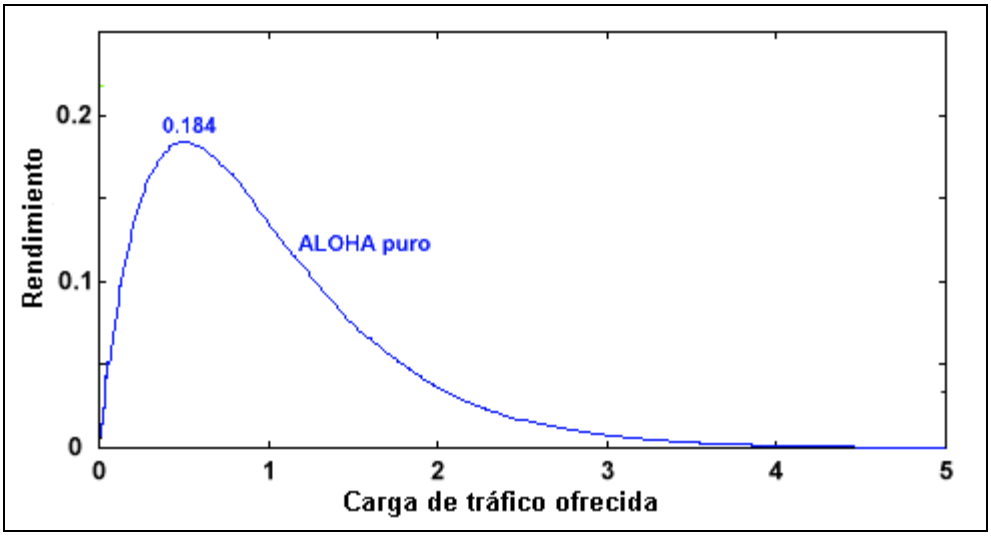


Figura 4-3 Eficiencia de Aloha Puro

4.2. DESARROLLO DE LA EVALUACIÓN

A continuación validaremos mediante simulación el modelo analítico del protocolo Aloha, estableciendo inicialmente un escenario satelital de simulación adecuado para tal fin.

El escenario de simulación utilizado en este trabajo consistió en una red de estaciones terrenas distribuidas en una recta ubicada en la intersección del meridiano de Greenwich y el paralelo Ecuatoriano, además de un satélite geoestacionario trabajando como repetidor con una capacidad de canal de 2Mb/s. Esto se muestra en la Figura 4-4.

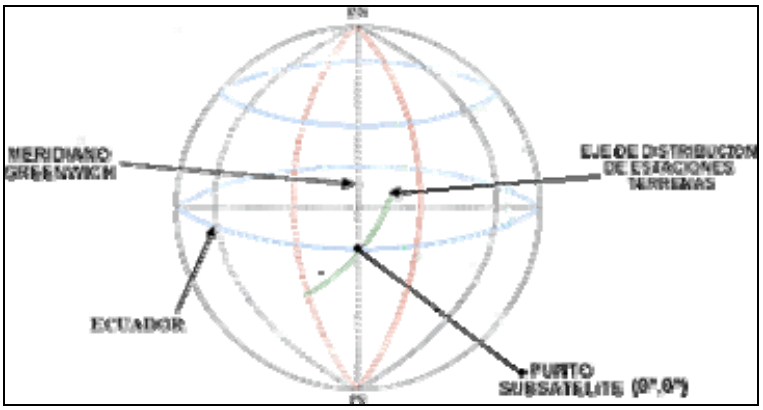


Figura 4-4 Distribución de estaciones terrenas y ubicación del satélite

4.2.1. LIMITACIONES

Cabe aclarar que una simulación precisa de una red satelital requiere de modelos detallados de radiofrecuencia, interacción de protocolos o efectos de fuerzas perturbadoras en las orbitas. Sin embargo, con el propósito de estudiar las características fundamentales de las redes satelitales desde una perspectiva de red, ciertos rasgos pueden ignorarse para el análisis.

4.2.2. MÉTRICAS

Una vez establecido el sistema y sus limitaciones, nos queda determinar los criterios para cuantificar el rendimiento del sistema y comparar sus prestaciones. Primero evaluaremos el sistema con base en el arribo de paquetes exitosos. De lo anterior, mediremos el ancho de banda ocupado en cada caso, así como el retardo del sistema.

4.2.3. PARÁMETROS

Una vez definidos los parámetros de desempeño a evaluar, es necesario identificar las variables que se emplearán en el entorno de simulación. A continuación, se describen las variables utilizadas en los scripts de simulación para el protocolo Aloha:

- **mean_backoff**: Coeficiente inicial del tiempo aleatorio cuando existe retransmisión.
- **rtx_limit**: Numero máximo de intentos de retransmisión.
- **send_timeout**: Tiempo de retransmisión cuando no hay respuesta.
- **packetSize_**: Define el tamaño constante de los paquetes generados.
- **burst_time_**: Es el tiempo promedio “On” para el generador de paquetes. También conocido como tiempo de ráfaga.
- **idle_time_:** Es el tiempo promedio “Off” para el generador de paquetes.
- **rate_**: Es la tasa de envío durante los tiempos de “On”.
- **Número de Estaciones.**

4.2.4. SELECCIÓN DE FACTORES

Recordando que los factores son las variables seleccionadas que varían durante la evaluación del desempeño, elegimos el tamaño de los paquetes y el número de estaciones terrenas como nuestros factores, es decir, las variables que utilizaremos para llevar a cabo la evaluación, ya que estos influyen significativamente en el rendimiento del sistema.

Para esto, inicialmente vamos a variar el tamaño de los paquetes cada 100 bytes hasta un máximo de 1000 bytes, para una red con 100 estaciones terrenas. Lo anterior se repetirá para los casos de 150, 200 y 250 estaciones terrenas.

4.2.5. CARGA DE TRABAJO

En cada caso, la mitad de las estaciones terrenas serán transmisores y la otra mitad receptores. Cada estación transmisora generará un tráfico exponencial y utilizará el protocolo de transporte UDP.

4.3. DESARROLLO

Una vez establecido los escenarios de simulación, se realizaron las simulaciones considerando un tiempo de simulación de 100 segundos. La Tabla 4-1, muestra los resultados que se obtuvieron.

4.3.1. ANÁLISIS DE LOS DATOS

Con los datos obtenidos se realizó la evaluación del desempeño. Para esto, comenzamos con el cálculo del ancho de banda en cada escenario. Esto lo realizamos utilizando la siguiente ecuación.

$$BW = \frac{(\#PaqEnv + \#Colisiones) \times (PacketSize) \times 8}{Tiempo\_de\_Simulacion} = \left[b/seg \right]$$

Los resultados se reportan en la Tabla 4-2

Tabla 4-1 Datos obtenidos de la simulación

100 estaciones terrenas			
Packetsize (Bytes)	Enviados (Paquetes)	Recibidos (Paquetes)	Colisionados (Paquetes)
100	9888	9858	1071
200	9961	9915	2275
300	9893	9755	3832
400	9902	9620	5681
500	9535	8888	7873
600	8878	7859	9521
700	8153	6826	10551
800	7454	5853	11470
900	6801	4894	12135
1000	6356	4347	12288

150 estaciones terrenas			
Packetsize (Bytes)	Enviados (Paquetes)	Recibidos (Paquetes)	Colisionados (Paquetes)
100	14919	14862	2574
200	14746	14553	6066
300	14233	13400	11575
400	12613	11020	15111
500	11082	8877	16884
600	9936	7291	17979
700	9077	5949	18809
800	8365	4869	19529
900	7888	4085	19971
1000	7486	3325	20570

200 estaciones terrenas			
Packetsize (Bytes)	Enviados (Paquetes)	Recibidos (Paquetes)	Colisionados (Paquetes)
100	19958	19839	4920
200	19317	18478	14190
300	16652	14417	20663
400	14071	10848	23326
500	12244	8291	24707
600	11053	6394	25934
700	10249	4941	26980
800	9580	3867	27635
900	9158	3077	28227
1000	8784	2356	28627

250 estaciones terrenas			
Packetsize (Bytes)	Enviados (Paquetes)	Recibidos (Paquetes)	Colisionados (Paquetes)
100	24558	24320	8658
200	22287	20202	23340
300	18001	14223	28587
400	15168	10144	31018
500	13518	7393	33046
600	12403	5562	34111
700	11518	4040	34947
800	11004	2941	35830
900	10609	2233	36094
1000	10337	1594	36791

Tabla 4-2 Ancho de banda utilizado

100 estaciones terrenas			
Packetsize (Bytes)	Enviados (Paquetes)	Colisionados (Paquetes)	BW
100	9888	1071	87,672
200	9961	2275	195,776
300	9893	3832	329,400
400	9902	5681	498,656
500	9535	7873	696,320
600	8878	9521	883,152
700	8153	10551	1,047,424
800	7454	11470	1,211,136
900	6801	12135	1,363,392
1000	6356	12288	1,491,520

150 estaciones terrenas			
Packetsize (Bytes)	Enviados (Paquetes)	Colisionados (Paquetes)	BW
100	14919	2574	139,944
200	14746	6066	332,992
300	14233	11575	619,392
400	12613	15111	887,168
500	11082	16884	1,118,640
600	9936	17979	1,339,920
700	9077	18809	1,561,616
800	8365	19529	1,785,216
900	7888	19971	2,005,848
1000	7486	20570	2,244,480

200 estaciones terrenas			
Packetsize (Bytes)	Enviados (Paquetes)	Colisionados (Paquetes)	BW
100	19958	4920	199,024
200	19317	14190	536,112
300	16652	20663	895,560
400	14071	23326	1,196,704
500	12244	24707	1,478,040
600	11053	25934	1,775,376
700	10249	26980	2,084,824
800	9580	27635	2,381,760
900	9158	28227	2,691,720
1000	8784	28627	2,992,880

250 estaciones terrenas			
Packetsize (Bytes)	Enviados (Paquetes)	Colisionados (Paquetes)	BW
100	24558	8658	265,728
200	22287	23340	730,032
300	18001	28587	1,118,112
400	15168	31018	1,477,952
500	13518	33046	1,862,560
600	12403	34111	2,232,672
700	11518	34947	2,602,040
800	11004	35830	2,997,376
900	10609	36094	3,362,616
1000	10337	36791	3,770,240

Cuando el número de estaciones terrenas se incrementa, en conjunto con el incremento del tamaño de los paquetes, se observa que el ancho de banda requerido sobrepasa la capacidad de canal del sistema, la cuál es de 2 Mb/s. La Figura 4-5 muestra la grafica donde se compara el uso de ancho de banda de los cuatro casos.

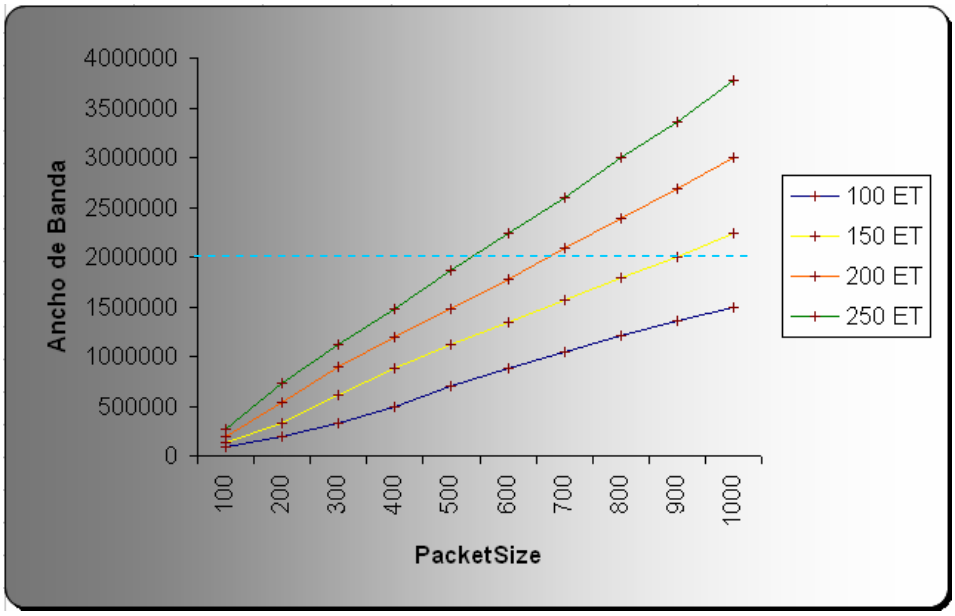


Figura 4-5 Ancho de Banda demandado en cada caso

Una vez obtenido el ancho de banda demandado en cada escenario, se relacionó con la cantidad de bits que arribaron exitosamente. Estos se obtienen a partir del número de paquetes recibidos.

La Tabla 4-3 nos muestra el promedio de la cantidad de bits recibidos exitosamente por segundo en todas las estaciones terrenas para cada uno de los casos. Después en la Figura 4-6, podemos observar las gráficas obtenidas para los casos de 100 y 150 estaciones terrenas, mientras que en la Figura 4-7 se observan las graficas para 200 y 250 estaciones terrenas. En las cuatro gráficas, se relaciona el ancho de banda total utilizado con la cantidad de bits que arribaron exitosamente.

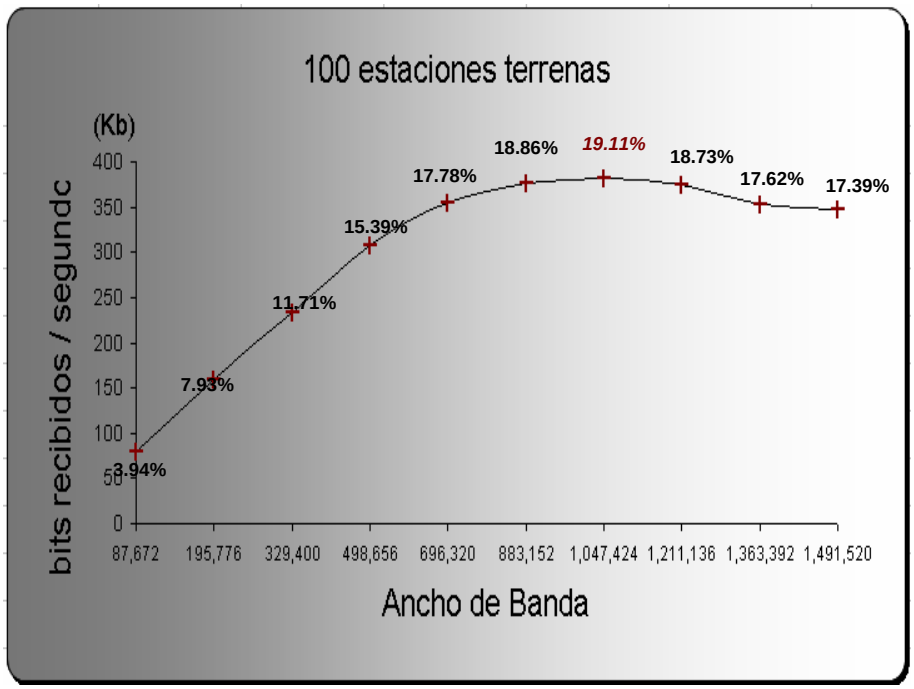
Tabla 4-3 Cantidad de arribo exitoso en bits

100 estaciones terrenas			
Packetsize (Bytes)	Recibidos (Paquetes)	Bits recibidos	Bits recibidos por segundo
100	9858	7886400	78864
200	9915	15864000	158640
300	9755	23412000	234120
400	9620	30784000	307840
500	8888	35552000	355520
600	7859	37723200	377232
700	6826	38225600	382256
800	5853	37459200	374592
900	4894	35236800	352368
1000	4347	34776000	347760

150 estaciones terrenas			
Packetsize (Bytes)	Recibidos (Paquetes)	Bits recibidos	Bits recibidos por segundo
100	14862	11889600	118896
200	14553	23284800	232848
300	13400	32160000	321600
400	11020	35264000	352640
500	8877	35508000	355080
600	7291	34996800	349968
700	5949	33314400	333144
800	4869	31161600	311616
900	4085	29412000	294120
1000	3325	26600000	266000

200 estaciones terrenas			
Packetsize (Bytes)	Recibidos (Paquetes)	Bits recibidos	Bits recibidos por segundo
100	19839	15871200	158712
200	18478	29564800	295648
300	14417	34600800	346008
400	10848	34713600	347136
500	8291	33164000	331640
600	6394	30691200	306912
700	4941	27669600	276696
800	3867	24748800	247488
900	3077	22154400	221544
1000	2356	18848000	188480

250 estaciones terrenas			
Packetsize (Bytes)	Recibidos (Paquetes)	Bits recibidos	Bits recibidos por segundo
100	24320	19456000	194560
200	20202	32323200	323232
300	14223	34135200	341352
400	10144	32460800	324608
500	7393	29572000	295720
600	5562	26697600	266976
700	4040	22624000	226240
800	2941	18822400	188224
900	2233	16077600	160776
1000	1594	12752000	127520



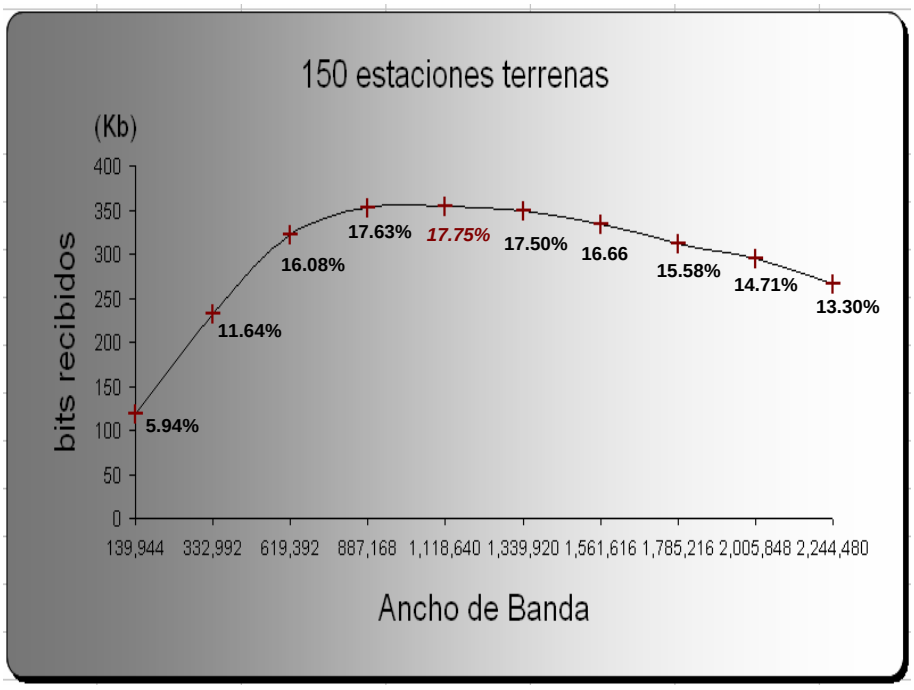
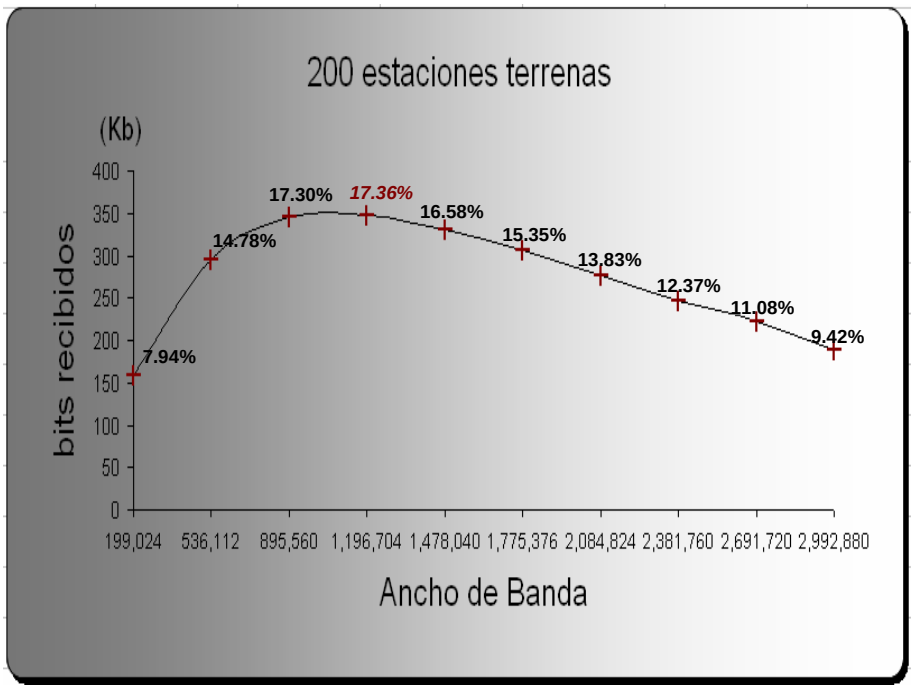


Figura 4-6 Graficas de rendimiento
para 100 y 150 estaciones terrenas



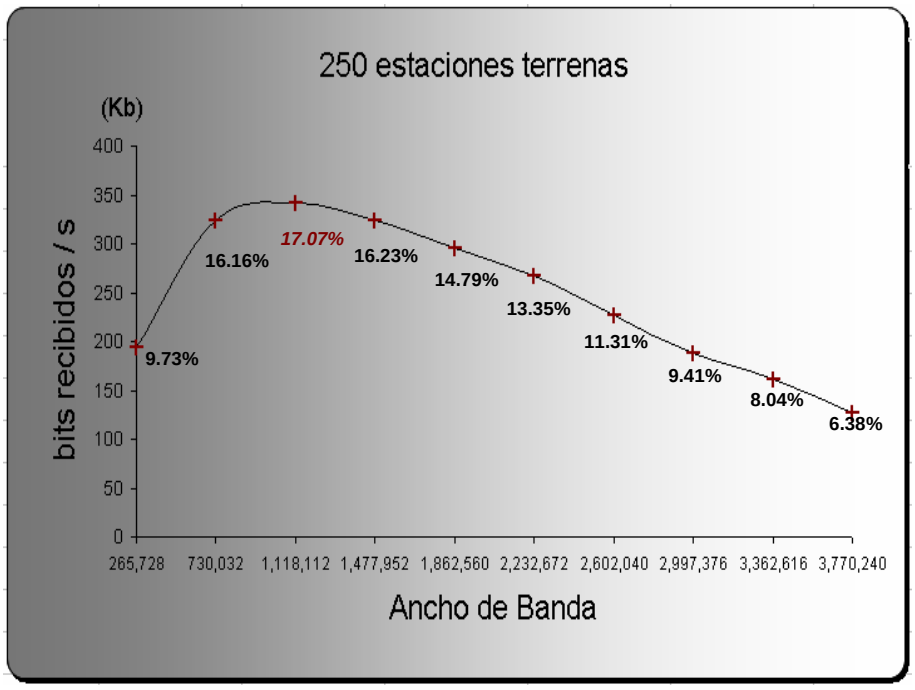


Figura 4-7 Gráficas de rendimiento para 200 y 250 estaciones terrenas

En las gráficas podemos observar que el valor máximo de bits recibidos exitosamente se encuentra alrededor del 18.4% de la capacidad máxima de la red, que es lo esperado a en modelo analítico.

En los casos donde existen 150, 200 y 250 estaciones terrenas el máximo *throughput* alcanzado se encuentra por debajo del teórico, estos máximos se encuentran en los puntos donde se utilizó un tamaño de paquete de 500, 400 y 300 Bytes respectivamente. De aquí se deduce que a mayor número de estaciones terrenas que interactúan con el protocolo Aloha el máximo *throughput* se alcanzará con un tamaño menor de paquete.

En el caso donde existen 100 estaciones terrenas el máximo *throughput* alcanzado es de 19.11% que correspondió con un tamaño de paquete de 700 Bytes. Aquí podemos observar que el *throughput* supera el valor teórico, lo cual sugiere que teniendo una baja cantidad de estaciones podemos transmitir paquetes de mayor tamaño que en los demás sistemas y así maximizar el rendimiento de la red.

4.3.2. TIEMPO DE RESPUESTA (Delay)

Es importante evaluar el retardo que existe en la transmisión de paquetes. A partir del ancho de banda donde se obtiene el máximo *throughput* se puede calcular el retardo promedio de cada caso. También calculamos el retardo que existe para el tamaño de paquete de 100 y 1000 Bytes para tomarlos como referencia. Para calcular el tiempo de respuesta solo requerimos de los siguientes parámetros.

- T_R= Tiempo de propagación
- τ = Duración del paquete
- K= Estaciones Transmisoras
- G= Throughput
- P.S. =Tamaño de paquetes

El cálculo del retardo en el sistema Aloha se obtuvo con la siguiente ecuación:

$$T_{Aloha} = T_R + \tau + \left[e^{2G} - 1\right]\left[T_R + \frac{(K+1)\tau}{2}\right]$$

La Tabla 4-4 muestra los resultados obtenidos para los cuatro casos:

Tabla 4-4 Tiempos de Retardo

100 estaciones terrenas		
Packetsize (Bytes)	Duración del paquete τ (mseg)	T_{Aloha} (mseg)
100	0.4	260.044
700	2.67028	765.70809
1000	4	1312.02

150 estaciones terrenas		
Packetsize (Bytes)	Duración del paquete τ (mseg)	T_{Aloha} (mseg)
100	0.4	274.55
500	1.90734	831.7
1000	4	3168.43

200 estaciones terrenas		
Packetsize (Bytes)	Duración del paquete τ (mseg)	T_{Aloha} (mseg)
100	0.4	292.36
400	1.52587	910.81
1000	4	7439.08

250 estaciones terrenas		
Packetsize (Bytes)	Duración del paquete τ (mseg)	T_{Aloha} (mseg)
100	0.4	314.3
300	1.14440	829.75
1000	4	17602.7

De la Tabla 4-4 podemos observar que, en los puntos donde se encuentra el máximo *throughput* el retardo promedio que existe es menor a un segundo y observando la Figura 4-8 podemos darnos cuenta que el retardo es incluso menor al que existe en sistemas que utilizan FDMA o TDMA.

En el caso donde hay un tamaño de paquete de 1000 Bytes con 100 estaciones terrenas tenemos un valor de retardo por debajo del que existe en FDMA y TDMA, para el caso de 150 estaciones y el mismo tamaño de paquete observamos que el retardo comienza a aproximarse al que esta presente en sistemas que manejan TDMA. Para el caso de 200 y 250 estaciones y el mismo tamaño de paquete antes mencionado el retardo es mayor y en este caso poco práctico.

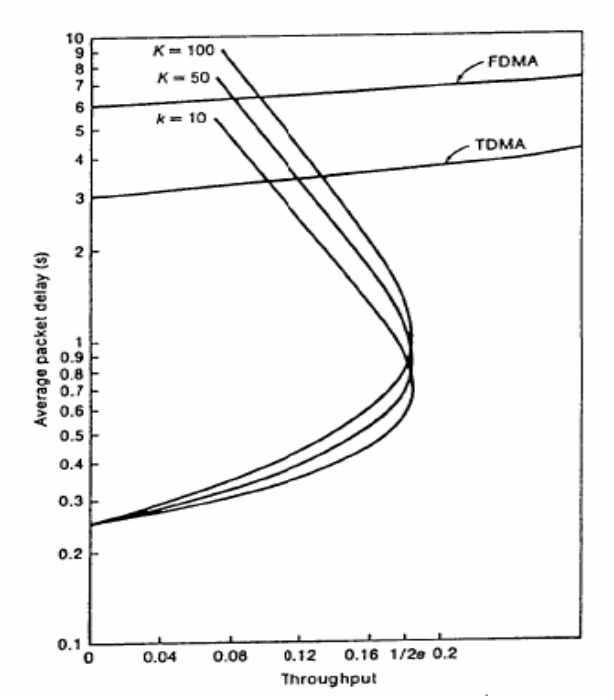


Figura 4-8 Grafica del Retardo Promedio vs throughput del protocolo Aloha con respesto a TDMA y FDMA.

4.3.3. PROBABILIDAD DE COLISIONES

También el simulador nos arroja datos para poder calcular la probabilidad de que haya colisiones de paquetes en el sistema, así como el número promedio de retransmisiones de un mismo paquete. Para realizar esto, tomamos de las tablas la siguiente información.

- Paquetes Recibidos
- Paquetes Colisionados
- Duración de Paquete

Las ecuaciones utilizadas son las siguientes:

$$\lambda' = \frac{\text{Paq.Recebidos} + \text{Paq.Colisionados}}{\text{tiempo de simulacion}}$$

$P_t = e^{-2\lambda'\tau}$ Probabilidad de que no haya colisión entre dos paquetes.

$N = e^{2\lambda'\tau}$ Numero promedio de retransmisiones de un paquete

La Tabla 4-5, reporta los resultados obtenidos:

Tabla 4-5 Probabilidad de no colisión y promedio de retransmisiones

100 estaciones terrenas		
Packetsize (Bytes)	P_t %	N
100	91.62	1.09
700	**39.53**	**2.52**
1000	26.42	3.78

150 estaciones terrenas		
Packetsize (Bytes)	P_t %	N
100	86.98	1.14
500	**37.42**	**2.67**
1000	14.78	6.76

200 estaciones terrenas		
Packetsize (Bytes)	P_t %	N
100	82.03	1.21
400	**35.24**	**2.83**
1000	8.38	11.92

250 estaciones terrenas		
Packetsize (Bytes)	P_t %	N
100	76.81	1.3
300	**37.53**	**2.66**
1000	10.32	9.68

De la Tabla 4-5 se observa que, en general, al tener un tamaño menor de paquete las probabilidades de que exista una colisión son reducidas. Los resultados también nos muestran que en el caso de máximo *throughput* va a existir una probabilidad de entre 35% y 40% de que no existan colisiones entre paquetes para los cuatro casos. Cuando existe un tamaño de paquete de 1000 Bytes la probabilidad de que el paquete colisione es mayor; particularmente, tomando el caso en donde existe 250 estaciones terrenas van a existir en promedio 10 transmisiones exitosas de cada 100 transmisiones hechas, en contraste con el

caso de 100 estaciones terrenas, donde el rendimiento es mejor para el mismo tamaño de paquete.

Finalmente, del cálculo de posibles retransmisiones realizado para los cuatro casos, los resultados nos muestran que en el rango del máximo *throughput*, un paquete hará el intento de ser retransmitido en promedio de dos a tres veces antes de ser recibido con éxito. En el caso de un tamaño mayor de paquete, los resultados nos arrojan que se necesitaría un número mayor de intentos de retransmisión, pero cabe aclarar que en la simulación el número de intentos de retransmisión esta limitado a tres intentos, esto se puede comprobar en el *script* **(apéndice B)**.

CONCLUSIONES

En este trabajo desarrollamos la evaluación del protocolo Aloha en un escenario satelital, gracias a la utilización del simulador de redes Network Simulator 2. Esta evaluación parte del modelo teórico del protocolo Aloha y fue verificado con los resultados arrojados de las simulaciones.

Para esto, desarrollamos un conjunto de escenarios y llevamos a cabo varias simulaciones observando qué parámetros influían más en el desempeño del protocolo, descubriendo que era el tamaño de paquetes y el número de estaciones. Al encontrar los parámetros marcamos el número de pruebas que requeríamos para realizar la evaluación. Cabe aclarar que caso de querer evaluar un número mayor a 250 estaciones terrenas se requeriría un ordenador con mayores prestaciones para realizar las simulaciones.

Con los resultados arrojados por el simulador, podemos darnos cuenta que, conforme el numero de estaciones aumenta obtenemos la mayor eficiencia del sistema al disminuir el tamaño de los paquetes. Las gráficas obtenidas con los resultados del simulador muestran un comportamiento muy similar a la grafica que se obtiene teóricamente del protocolo Aloha, en donde al momento de ocupar el 50% del total del ancho de banda se obtiene la mayor eficiencia del sistema en los cuatro escenarios. Cabe aclarar que, aunque el ancho de banda exceda el 50%, la eficiencia no decrece drásticamente. A raíz de estos valores pudimos comparar los resultados con el modelo analítico del protocolo Aloha por lo que podemos asegurar la efectividad de esta evaluación con esta herramienta de simulación.

Para el caso del retardo, el resultado más óptimo se encuentra en el escenario con menor número de estaciones transmitiendo, y este fue aumentando conforme aumentaba el número de estaciones, sin embargo cabe destacar que en el escenario donde tenemos 250 estaciones se observa un retardo menor que el de los escenarios de 150 y 200 estaciones.

En los cálculos de la probabilidad de no colisión y del número promedio de retransmisiones, se observa que al aumentar el número de estaciones, el número promedio de retransmisiones crece gradualmente, mientras que la probabilidad de que no haya colisión y el promedio de retransmisiones, al momento de transmitir un paquete, tiende a permanecer constante para los cuatro casos en donde se obtuvo la mayor eficiencia.

Las simulaciones se realizaron utilizando el NS-2, lo que ha supuesto una dificultad añadida.

Con esto podemos recalcar la importancia de realizar evaluaciones con herramientas de simulación a los sistemas informáticos ya que con esto podemos obtener una visión más detallada en comparación con el modelado y más práctica que la medición. Además gracias a estas herramientas de simulación, observamos con mayor facilidad qué características del protocolo impactaban en el rendimiento.

PROPUESTA PARA TRABAJOS FUTUROS

En primer lugar este trabajo podría ser punto de partida para otros trabajos de evaluación o desarrollo de algún protocolo. En este trabajo se evaluó el desempeño del protocolo Aloha pero se podría llevar a cabo otras evaluaciones tal como podría ser una evaluación al protocolo Aloha-ranurado o incluso la evaluación de otros protocolos de acceso al medio en comunicaciones satelitales.

También cabe aclarar una de las ventajas de Net Simulator 2, que al ser software libre se puede desarrollar módulos adicionales donde se puede incluir o desarrollar nuevos protocolos de gestión de redes y también adquirir experiencias de otros módulos desarrollados por la comunidad que utiliza este simulador.

APÉNDICE A: INSTALACIÓN DEL NS-2

El NS-2 es un simulador que fue programado para trabajar en plataformas similares a UNIX (FreeSBD, Linux, SunSO, Solaris), por lo que la instalación resulta más sencilla en ellos, sin embargo es posible utilizarlo en otros sistemas operativos, mientras se cuente con un compilador C++. Para trabajar en Windows, es necesario el programa Cygwin, el cual provee de un ambiente similar a Linux, aun así, el NS-2 puede generar problemas al momento de su instalación y validación en este S.O.

Este simulador es bastante amplio, el paquete completo (all-in-one) tiene un tamaño aproximado de 320 MB, sin embargo se pueden instalar solo aquellos paquetes con los que se desea trabajar. En ésta sección nos limitaremos a la instalación del paquete all-in-one sobre una plataforma LINUX (Mandriva 2006). *Si desea conocer los pasos para la instalación por partes, o instalación en Windows por favor visite la página oficial.*

INSTALACIÓN Y CONFIGURACIÓN

El paquete All-in-one se puede descargar desde la página oficial del simulador *(www.isi.edu/nsnam/ns)* en su versión más reciente. Este paquete contiene los componentes requeridos y opcionales usados en la ejecución del NS-2. Este paquete también cuenta con un ejecutable "Install" el cual automáticamente configura, compila e instala estos componentes.

El archivo descargado es un archivo comprimido, los archivos deben extraerse utilizando el gestor de archivos comprimidos del Sistema Operativo (Ark en el caso de Mandriva 2006).

Una vez extraídos los archivos, abra la consola de comandos, diríjase a la carpeta donde fueron extraídos los archivos, y digite el siguiente comando: *"./install"*. Este

comando se encarga de configurar y relacionar todas las carpetas necesarias para la ejecución del NS-2.

Terminado el proceso anterior, entre a la carpeta de "*…/ns-allinone/ns-2.31" (o de la versión que haya instalado)*, desde la consola de comando digite el comando: *"./validate"*, esto iniciara el proceso de validación del NS-2, y su objetivo es revisar que todos los paquetes funcionen correctamente, este proceso dura varios minutos en ser finalizado.

El sistema operativo Linux permite la creación de comandos generales, es decir, se puede llamar a un ejecutable desde cualquier carpeta mientras se encuentre en la consola de comandos. Esto resulta muy útil ya que nos permitirá ejecutar el NS-2 y todas sus funciones desde cualquier punto, facilitando el manejo y la administración de nuestras simulaciones.

Normalmente para crear este tipo de función se utiliza la variable de entorno *$PATH,* que se utiliza con una serie de instrucciones que resultan complicadas para los usuarios que no están familiarizados con Linux. Afortunadamente el NS-2, en el momento en que se instala, crea una serie de archivos que se guardan en la carpeta *"…/ns-allinone/bin/",* estos contienen los *PATH* que apuntan a los ejecutables utilizados en el NS-2. Estos archivos se deben copiar a la carpeta *"//BIN/"* que se encuentra en la raíz del sistema operativo (se recomienda hacerlo desde la cuenta de administrador "*root"*). Esto finaliza la configuración y el NS-2 está listo para utilizarse.

APÉNDICE B: SCRIPT ALOHA

```
# 1.  basic:  El control de acceso al medio opera en modo alto-espera (un paquete
#       en  tiempo de espera).  Los paquetes colisionados y los arrojados no van a
#       ser trazados.  Esta prueba debe proveer resultados similares al conjunto
#       de pruebas para satélite.

# 2.  basic_tracing: Tiene un funcionamiento similar al tipo "basic", pero los
#       paquetes colisionados y los arrojados son trazados explicitamente.
#
# 3.  poisson: Los paquetes arriban de acuerdo a la distribucion de Poisson. Cada
#               fuente opera en modo alto-espera y tanto los paquetes colisionados
#               como los arrojados son trazados. Cuando rtx_limit_ == 0 (es el modo
#                 no persistente).
#               Esto puede usarse para tratar de obtener los resultados aproximados
#               de Aloha puro, si el numero de terminales es muy grande comparado
#               con la tasa de transferencia (para que no halla encolamiento de
#               paquetes).

if { $argc != 1 } {
       puts stderr {usage: ns sat-aloha.tcl [basic basic_tracing poisson]}
       exit 1
}
set test_ $argv
puts "Running test $test_ ..."

global ns
set ns [new Simulator]

# Cofiguracion general para el protocolo Aloha
Mac/Sat/UnslottedAloha set mean_backoff_ 0.5s ; # mean exponential backoff
time(s)
Mac/Sat/UnslottedAloha set rtx_limit_ 3; # numero maximo de intentos de
#retransmision
Mac/Sat/UnslottedAloha set send_timeout_ 270ms; # tiempo de reenvio si no hay
#respuesta

if { $test_ == "basic"} {
        Mac/Sat set trace_collisions_ false
        Mac/Sat set trace_drops_ false
}

global opt
set opt(chan)          Channel/Sat
```

```
set opt(bw_up)                2Mb
set opt(bw_down)    2Mb
set opt(phy)            Phy/Sat
set opt(mac)            Mac/Sat/UnslottedAloha
set opt(ifq)            Queue/DropTail
set opt(qlim)           50
set opt(ll)             LL/Sat
set opt(wiredRouting)   OFF

#Creación de archivo de Traza
set outfile [open out.tr w]
$ns trace-all $outfile

# Configuración del satélite y de los nodos terrestres

# GEO satélite ubicado en longitud igual a 0 grados
$ns node-config -satNodeType geo-repeater \
                -llType $opt(ll) \
                -ifqType $opt(ifq) \
                -ifqLen $opt(qlim) \
                -macType $opt(mac) \
                -phyType $opt(phy) \
                -channelType $opt(chan) \
                -downlinkBW $opt(bw_down) \
                -wiredRouting $opt(wiredRouting)
set n1 [$ns node]
$n1 set-position 0

# Configuración de las terminales y sus diferentes localizaciones
$ns node-config -satNodeType terminal
set num_nodes       100
for {set a 1} {$a <= $num_nodes} {incr a} {
        set n($a) [$ns node]
        $n($a) set-position [expr -37.5 + $a * 0.3] [expr 37.5 - $a * 0.3]
        $n($a) add-gsl geo $opt(ll) $opt(ifq) $opt(qlim) $opt(mac) $opt(bw_up) \
                $opt(phy) [$n1 set downlink_] [$n1 set uplink_]
}

for {set a 1} {$a <= $num_nodes} {incr a} {
        set b [expr int($a + (0.5 * $num_nodes))]
        if {$b > $num_nodes} {
                incr b [expr -1 * $num_nodes]
        }

        set udp($a) [new Agent/UDP]
```

```
        $ns attach-agent $n($a) $udp($a)
        set exp($a) [new Application/Traffic/Exponential]
        $exp($a) attach-agent $udp($a)

        $exp($a) set rate_ 10000Mb
        $exp($a) set burst_time_ 0
        $exp($a) set idle_time_ 5s
        $exp($a) set packetSize_ 1000
        if {$test_ == "poisson"} {

                $exp($a) set rate_ 10000Mb
                $exp($a) set burst_time_ 0
                $exp($a) set idle_time_ 1s
        }

        set null($a) [new Agent/Null]
        $ns attach-agent $n($b) $null($a)

        $ns connect $udp($a) $null($a)
        $ns at 1.0 "$exp($a) start"
}

$ns trace-all-satlinks $outfile

# Se recomienda el uso de ruteo Centralizado
set satrouteobject_ [new SatRouteObject]
$satrouteobject_ compute_routes

$ns at 100.0 "finish"

proc finish {} {
        global ns outfile
        $ns flush-trace
        close $outfile

        exit 0
}

$ns run
```

BIBLIOGRAFÍA

[1] Tri T. Ha. Digital Satellite Communication. Mc Graw Hill.1990

[2] L. Mercader del Rio Comunicaciones por Satélite. Apuntes de Programa de Postgrado en Sistemas y Redes de Comunicaciones, UPM.

[3] Slark Digital Comunication. Prentice-Hall. 1988

[4] Raj Jain. The art of computer systems performance analysis : techniques for experimental design, measurement, simulation, and modeling. Wiley, New York,1991.

[5] Uc, Berkely, LBL, USC/ISI, ans Xerox PARC. The ns Manual, April 2002.

[6] Jae Chung and Mark Claypool. NS by example. Worcester Polytechnic Institute,Computer Science.

[7] Roque Saens Peña Teoría de las Telecomunicaciones B1876BXD Universidad Nacional de Quilmas.

[8] María Eugenia Carvente González y Alina Adriana Macias Peral EVALUACIÓN DE LA CALIDAD DE SERVICIO DE UNA RED DE DATOS CON TRÁFICO IP, Tesis de Ingeniería en Comunicaciones y Electrónica, E.S.I.M.E. IPN 2006.

[9] IEEE Electrical Engineering Dictionary 2000 CRC.

[10] Wayne Tomasi Sistemas de Comunicaciones Electrónicas. Prentice-Hall 2003

[11] Comunicaciones por Satélite Rodolfo Neri Vela. Thomson 2003