ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

"IMPLEMENTACION DE UNA HERRAMIENTA DIDACTICA PARA LA SIMULACION DE REDES BLUETOOTH ORIENTADA A PRACTICAS EN EL LABORATORIO DE TELECOMUNICACIONES"

TESIS DE GRADO

Previa a la obtención del Título de:

INGENIERO EN ELECTRONICA Y TELECOMUNICACIONES

Presentado por

JOSE LUIS LANDA ANZULES

JOSE LUIS JIMBO SALAZAR

LUIS FERNANDO SALAZAR MARTRUZ

Guayaquil – Ecuador

2007

AGRADECIMIENTO

Agradecemos a nuestras familias por el apoyo que siempre nos brindaron y supieron estar ahí cuando los necesitamos, a la Ing. Rebeca Estrada, a Dios y a todos los que hicieron posible que se desarrolle esta tesis.

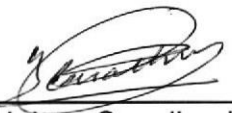
DEDICATORIA

A nuestros padres y hermanos

A nosotros

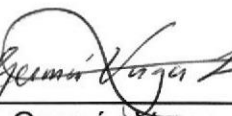
A nuestra directora de Tesis

TRIBUNAL DE GRADO



Ing. Holger Cevallos U.
SUBDECANO DE LA FIEC

Ing. Rebeca Estrada
DIRECTOR DE TESIS



Ing. Germán Vargas
VOCAL PRINCIPAL

Ing. Washington Medina
VOCAL PRINCIPAL

DECLARACION EXPRESA

"La responsabilidad del contenido de esta Tesis de Grado, nos corresponden exclusivamente; y el patrimonio intelectual de la misma a la Escuela Superior Politécnica del Litoral".

José Landa Anzules

José Jimbo Salazar

Luis Salazar Martruz

RESUMEN

Movidos por la ausencia de herramientas que nos ayuden a apreciar de forma gráfica el funcionamiento de la tecnología inalámbrica Bluetooth, decidimos desarrollar simulaciones de escenarios frecuentes de este estándar, así podríamos observar, entender y aprender como se lleva a cabo la comunicación entre dispositivos utilizando este tipo de redes. Estas simulaciones fueron creadas como herramientas didácticas, de tal forma, que los centros educativos las utilicen para los estudiantes de pre-grado y post-grado de carreras afines a las telecomunicaciones.

Nuestro trabajo se deriva del desarrollo de la librería para la simulación de redes Bluetooth de la universidad de Cincinnati (UCBT) que a su vez es compatible con el simulador de redes NS2. En resumen hemos planteado una adaptación de esta librería para que trabaje con el Network Animator (NAM), dándole un ambiente gráfico a los procesos que se dan dentro de las operaciones de los dispositivos Bluetooth. El propósito de esta adaptación de la librería UCBT es conseguir una herramienta gráfica fiable de simulación de los casos más frecuentes de la tecnología Bluetooth.

INDICE GENERAL

Capitulo 3:

3. Descripción del proyecto

Capitulo 4:

4. *Implementación del Proyecto*

INDICE DE FIGURAS

INDICE DE TABLAS

ABREVIATURAS

Estas son las abreviaturas que utilizaremos en el presente documento, a lado el significado de cada una:

2D, 3D: Dos dimensiones, tres dimensiones

A2DP: Advanced Audio Distribution Profile, *perfil de distribución de audio avanzado*

ACL: Asynchronous Connectionless Link, *enlace asincrónico no orientado a la conexión*

ACL - C: ACL control logical link, *enlace ACL lógico de control*

ACL - U: ACL User logical link, *enlace ACL lógico de usuario*

AD – HOC: Red que se crea espontáneamente. No requiere una infraestructura fija y está limitada en el espacio y en el tiempo

AODV: Ad - Hoc On Demand Distance Vector, *protocolo de enrutamiento por vector distancia bajo demanda para redes Ad - Hoc*

ASB: Active Slave Broadcast, *difusión del dispositivo esclavo activo*

AVCTP: Audio / Video Control Transport Protocol, *protocolo de transporte para el control de audio y video*

AVRCP: Audio / Video Remote Control Profile, *perfil de control remoto de audio y vídeo*

BBP: Basic Printing Profile, *perfil básico de impresión*

BIP: Basic Imaging Profile, *perfil básico de imagen*

BNEP: Bluetooth Network Encapsulation Protocol, *protocolo de encapsulamiento de redes bluetooth*

C: Lenguaje de programación C

C++ / OTCL: Lenguaje de programación C++ / object - oriented Tool Command Language

CDMA: Code division multiple access, *acceso múltiple por división de código*

CIP: Common ISDN profile, *perfil común de ISDN*

CTP: Cordless Telephony Profile, *perfil de telefonía inalámbrica*

DRR: Deficit Round Robin

DSDV: Destination - Sequenced Distance Vectoring, *protocolo de enrutamiento dinámico de destinos en secuencia y distancia de vectores*

DSR: Dynamic Source Routing, *protocolo de enrutamiento de fuentes dinámicas*

DUN: Dial - up Networking Profile, *perfil para redes Dial - Up*

EIATIA - 232 - E: Electronic Industries Alliance / Telecommunications Industry Association 232 - E (Nombre actual del RS - 232)

Enlaces "C": enlaces de control

Enlaces "S": enlaces de ráfagas

Enlaces "U": enlaces de usuario

eSCO: Extended Synchronous Connection – Oriented, *SCO extendido*

ESDP: Extended Service Discovery Profile, *perfil de descubrimiento de servicios extendido*

ETSI: European Telecommunications Standards Institute, *Instituto europeo de estándar de telecomunicaciones*

FAX: Fax Profile, *perfil de fax*

FH: Frequency Hopping, *salto de frecuencia*

FHS: Frequency Hopping Synchronization, *sincronización de salto de frecuencia*

FTP: File Transfer Profile, *perfil de transferencia de archivos*

GAP: Generic Access Profile, *perfil de acceso genérico*

GAVDP: General Audio/Video Distribution Profile, *perfil de distribución genérica de audio y video*

GB: Gigabyte

GHz: Gigahertz

GOEP: Generic Object Exchange Profile, *perfil genérico de intercambio de objetos*

HCI: host controller interface, *interfaz de comandos entre el controlador de la banda base y el gestor de enlaces*

HCRP: Hard Copy Cable Replacement Profile, *perfil de sustitución de cable de copia impresa*

HFP: Hands - Free Profile, *perfil manos libres*

HID: Human Interface Device Profile, *perfil de dispositivo de interfaz humana*

HSP: Headset Profile, *perfil de auricular*

IBM: International Business Machines

ICP: Intercom Profile, *perfil de intercomunicador*

IEEE: Institute of Electronic and Electrical Engineering

IETF: Internet Engineering Task Force, *fuerza de tarea de ingeniería de internet*

IrDA: Infrared Data Association

ISDN: Integrated Services Digital Network profile, *perfil de red digital de servicios integrados*

IT: Information Technologies, *Tecnologías de información*

Kbps: Kilobits por segundo

KBs: Kilobytes por segundo

L2CAP: Logical Link Control and Adaptation Protocol, *Protocolo de control y adaptación de enlaces lógicos*

LAN: Local Area Network, *Red de área local*

LBNL: Laboratorio Nacional Lawrence Berkeley

LC: Link Controller, *controlador de enlace*

LL: Logical Link, *enlace lógico*

LM: Link Manager, *administrador de enlace*

LMP: Link Management Protocol, *protocolo de administración de enlaces*

MAC: Media Access Control, *control de acceso al medio*

MANET: Mobile Ad - Hoc Networking, *redes móviles Ad - Hoc*

MB: Megabyte

MC link: Enlace de radio de corto alcance

MHz: Megahertz

MIT: Massachusetts Institute of Technology

NAM: Network Animator

NS2: Network Simulator 2

OBEX: Object Exchange, *protocolo de intercambio de objetos*

OLSR: Optimized LinkState Routing Protocol, *protocolo de enrutamiento proactivo optimizado*

OPP: Object Push Profile, *protocolo de empuje del objeto*

PAN: Personal Area Network, *red de área personal*

PC: Personal Computer

PCS: Personal Communications Service, *Servicio personal de comunicaciones*

PMP: Participant in Multiple Piconets, *Participante en varias piconets*

PDU: Protocol Data Units, *Protocolo de unidad de datos*

PSB: Pasive Slave Broadcast, *difusión de esclavos pasivos*

QoS: Quality of service, *calidad de servicio*

RED: Random early detection, *detección anticipada aleatoria*

RF: Radio frequency

RFCOMM: Radio Frequency Communication

RS - 232: Recommended Standard - 232

Rt: Route, *Ruta*

SAP: SIM Access Profile, *Perfil de Acceso SIM*

SCO: Synchronous Connection – Oriented, *Sincrónico orientado a la conexión*

SDAP: Service Discovery Application Profile, *perfil de aplicación de descubrimiento de servicio*

SDP: Service Discovery Protocol, *protocolo de descubrimiento de servicio*

SIG: Special Interest Group, *Grupo de interés especial de Bluetooth*

SPP: Serial Port Profile, *perfil de puerto de serie*

SYNC: Synchronization Profile, *perfil de sincronización*

TCP/IP: Transmission Control Protocol / Internet protocol, *protocolo de control de transmisión / protocolo de internet*

TCS: Telephony Control Specification, *especificación de control de telefonía*

TORA: Temporally - Ordered Routing Algorithm, *algoritmo de enrutamiento de orden temporal*

TS 07.10: Especificacion 07.10

UCBT: Librería Bluetooth de la universidad de Cincinnati

UDP: User Datagram Protocol, *Protocolo de datagrama de usuario*

VDP: Video Distribution Profile, *Perfil de distribución de video*

WAP: WAP Over Bluetooth Profile, *Perfil WAP sobre Bluetooth*

INTRODUCCION

Bluetooth es un estándar de comunicación inalámbrica de bajo costo y consumo de energía mínimo, que se encuentra aún en desarrollo, para aplicaciones de corto alcance. Dado que esta tecnología aún sigue en perfeccionamiento, una herramienta poderosa para probar diseños y algoritmos de enrutamiento en redes Bluetooth es el simulador de redes NS2.

Este trabajo está compuesto de cuatro capítulos. En el primer capítulo se describe el problema existente en lo que respecta al análisis y simulación de redes Bluetooth, debido a la ausencia de una herramienta pedagógica para la simulación de este tipo de redes inalámbricas orientada a los estudiantes de la carrera de Telecomunicaciones, así como estudiantes de otras carreras con conocimientos de redes de datos.

Dentro del ámbito de los simuladores el principal problema es la interpretación de los resultados, para lo cual hemos adaptado una librería para facilitar el uso de las salidas técnicas y visualizar los eventos. Esto lo hacemos utilizando el simulador y la librería que seleccionamos para el presente proyecto.

En el segundo capítulo detallamos la tecnología Bluetooth, su historia, arquitectura, protocolos, perfiles y los procedimientos de conexión. Además

una breve descripción de las redes Ad - Hoc, sus fundamentos y la configuración de redes Bluetooth.

En el tercer capítulo describimos el funcionamiento de nuestro proyecto así como todas las herramientas y modificaciones que fueron necesarias para su implementación. Las herramientas que escogimos son: el simulador de redes NS2 para la simulación de redes inalámbricas, la librería Bluetooth de la Universidad de Cincinnati (UCBT) para el trabajo con nodos Bluetooth y el Network Animator (NAM) para la visualización de los resultados.

En el cuarto capítulo detallamos las simulaciones de los casos más frecuentes dentro de una red Bluetooth utilizando la librería UCBT con sus respectivas modificaciones. Además una breve descripción de los requerimientos y los costos de implementación del proyecto.

En los apéndices se encontrarán los manuales básicos de NS2, UCBT y NAM, así como las guías prácticas para los estudiantes y sus respectivos manuales para los tutores.

CAPÍTULO 1:

1. ANTECEDENTES

1.1 IDENTIFICACIÓN DEL PROBLEMA

En la Facultad de Ingeniería en Electricidad y Computación hemos notado la ausencia de una herramienta de simulación para el análisis de cierto tipo de redes y tecnologías de actualidad tales como son:

- Redes de Infraestructura
    - Wireless LAN
    - GSM/GPRS
    - CDMA
    - UMTS
- Redes Ad - Hoc
    - Wireless LAN
    - Bluetooth

Estos tipos de redes son actuales y están presentes en nuestro medio, es así que nos hemos visto en la obligación de buscar una solución en lo que respecta al análisis y funcionamiento de este tipo de tecnologías. El presente proyecto está dirigido únicamente hacia redes Ad - Hoc Bluetooth.

1.1.1 TECNOLOGÍA UTILIZADA

El presente proyecto está enfocado hacia la simulación de redes Bluetooth, para esto hemos escogido dentro de los simuladores que existen a nuestra disposición el simulador Network Simulator versión 2 (NS2), y como bases para la simulación de redes Bluetooth la librería desarrollada por la Universidad de Cincinnati para Bluetooth (UCBT).

Para esto necesitamos un PC Pentium 4 de 3.0 GHz o equivalente, memoria RAM 512MB o superior, espacio en disco duro de 4 GB como mínimo, que tenga instalado el sistema operativo Linux con todos los paquetes de herramientas de desarrollo, editores y la interfaz gráfica GNOME. Se puede trabajar de 2 maneras, usando directamente Linux en un arranque simultáneo al de Windows o en un emulador. En el caso del arranque doble, puede usarse un procesador Pentium 3 de 800 MHz o superior. Si se decide utilizar una maquina virtual "Virtual Workstation" se prefiere que sea procesador Pentium 4 de 3.0 GHz o superior.

1.1.2 LIMITACIONES OPERACIONALES

Este proyecto tiene ciertas limitaciones en su estructura, como se mencionó anteriormente, entre nuestros objetivos se encuentra el proporcionar esta herramienta para el análisis y desarrollo de redes Bluetooth, además de promover que este proyecto sea mejorado y corregido en caso de ser necesario. Teniendo esto en claro procedemos a clasificar el tipo de limitaciones del presente proyecto de la siguiente manera:

A. Limitaciones del Simulador

- Los resultados de las simulaciones suelen tener cambios debido a las variables aleatorias que el simulador maneja.
- Los recursos que maneja el simulador tienen varias dependencias que usualmente se deben comprobar antes de poder utilizarlos.

B. Limitaciones de la Librería

- No permite simular una piconet de 8 elementos activos conectados al dispositivo dominante de la misma.
- El protocolo SDP no está completamente desarrollado para actuar con diversos tipos de servicios.
- Básicamente se utiliza para simulaciones que cumplan con el perfil PAN.

C. Limitaciones de la Visualización de Resultados

- Para toda simulación que implique más de 5 nodos será difícil distinguir los procesos individuales de cada nodo ya que si todas las transmisiones son simultáneas se confundirán las de un nodo con las de los demás.
- El NAM nos ofrece gráficos en 2 dimensiones, si bien es cierto el NS2 tiene previsto soportar topologías en 3 dimensiones, esta función aún no ha sido implementada.
- En las capas que se manejan por dispositivo, no todos los procesos son visualizados por el NAM, por lo que este se debe apoyar en anotaciones y etiquetas o incluso de colores para obtener mejores resultados.

D. Limitaciones del Proyecto culminado

- Dado que UCBT continúa en desarrollo y que el autor (Qihe Wang) sigue corrigiendo errores, nuestro proyecto es básicamente una adaptación para el uso del NAM, logrando así obtener una herramienta didáctica, con resultados lo más exactos posibles, a pesar de las limitaciones antes mencionadas.

1.2 SIMULADORES DISPONIBLES

En el ámbito de los simuladores de redes escogimos el NS2 y damos a continuación una tabla explicativa del por qué de esta decisión..

	Network Simulator 2	OPNET	NetSim
Documentación	Media	Alta	Poca
Flexibilidad	Muy Buena	Dado a que es producto propietario, no brinda soluciones mas q las propuestas por el producto	Media
Orientación	Simulación de diversos tipos de redes cableadas o inalámbricas	A empresas de diversos tipos en áreas como IT, networking, desarrollo y análisis y defensa	Situaciones de Ethernet, enrutamiento, paquetes Tcp / Ip
Lenguajes	C++ / OTCL	PROTO C	C
Sistema de Funcionamiento	Linux	Windows	Windows/ Linux
Nivel de Desarrollo	Medio - Alto	Alto	Medio
Costos	Software de licencia libre	Licencia Operativa	Licencia Libre
Lista de correo	Si	Support Center	No

Tabla1.1: *Comparación de simuladores de redes.*

Librerías disponibles para NS2:

A continuación detallamos las características de las librerías que encontramos para trabajar con el simulador escogido, el NS2.

	BlueHoc	Blue ware	UCBT
Estado de Desarrollo	Medio	Alto	Medio
Documentación	Buena	Buena	Poca, se da como referencia al Estándar

Bases Técnicas	Desde los Inicios de Bluetooth, 2001	Basado en BlueHoc	Basado en la especificación Bluetooth 1.0, 1.2 y 2.0
Orientación	piconets	Piconets y scatternets	Formación y solución de piconets y scatternets
Visualización	Nula	Intermedio	Poca
Esquema de Resultados	Archivos Trace	Archivos Trace	Archivos Trace y Out
Lista de correo	Si	No	Si

Tabla 1.2: *Librerías Bluetooth disponibles para NS2.*

1.3 JUSTIFICACION DE LOS CAMBIOS

La desventaja central de la librería BlueHoc es la definición inicial de los nodos que serán maestros o esclavos dentro de la red Bluetooth, lo que dificulta la simulación de casos especiales, además de no contar con un ambiente gráfico para las simulaciones. En lo que respecta a la librería Blueware, por su estado de desarrollo tiende a ser una de las mejores elecciones, sin embargo no proveía mucha información de la estructura interna y la manera de desarrollar y experimentar dentro de la misma.

Tales razones nos llevaron a escoger la librería UCBT ya que tenía un nivel de resultados aceptables, pero le hacía falta un ambiente gráfico desarrollado en el NAM, el cual trabaja en conjunto con NS2. De esta forma, aparte de realizar las prácticas (simulaciones de los casos más frecuentes dentro de las redes Bluetooth), nos vimos en la tarea de desarrollar e incluir dentro de la librería algunas variables y líneas de código para obtener ciertas ventajas al utilizar el NAM.

CAPÍTULO 2:

2. TEORÍA

2.1 BLUETOOTH

Bluetooth es una especificación que define un estándar global de comunicación inalámbrica, que posibilita la transmisión de voz y datos entre diferentes equipos mediante un enlace por radiofrecuencia. Los principales objetivos que se pretende conseguir con esta especificación son:

- Facilitar las comunicaciones entre equipos móviles y fijos.
- Eliminar cables y conectores entre éstos.
- Ofrecer la posibilidad de crear pequeñas redes inalámbricas y facilitar la sincronización de datos entre equipos personales.

La tecnología Bluetooth comprende hardware, software y requerimientos de interoperatibilidad, por lo que para su desarrollo ha sido necesaria la participación de los principales fabricantes de los sectores de las telecomunicaciones y la informática, tales como: Ericsson, Nokia, Toshiba, IBM, Intel y otros. Posteriormente se han ido incorporando muchas más compañías, y se prevé que próximamente lo hagan también empresas de sectores tan variados como: automatización industrial, maquinaria, ocio y

entretenimiento, fabricantes de juguetes, electrodomésticos, etc., con lo que en poco tiempo se nos presentará un panorama de total conectividad de nuestros dispositivos tanto en casa como en el trabajo.

Los dispositivos Bluetooth operan en una frecuencia de radio que se encuentra entre los 2.4 y 2.48 GHz, con la posibilidad de realizar hasta 1.600 saltos por segundo entre 79 frecuencias, en intervalos de 1 MHz. Cada unidad incluye una radio, un controlador de enlaces de banda base y el software para la administración de los enlaces y flujo de datos.

Los usuarios tienen la opción de dos potencias de señal: un nivel de baja potencia para distancias de hasta 10 metros, y un nivel de alta potencia de hasta 100 metros de distancia para los puntos de acceso. Los dispositivos Bluetooth pueden conectarse simultáneamente hasta siete dispositivos más, sin incluir el dispositivo maestro. La velocidad máxima de transferencia de datos es de aproximadamente 720 Kbps por canal.

2.1.1 HISTORIA

El nombre de la tecnología Bluetooth se tomó de un rey danés del siglo X llamado Harald Blatand, cuya traducción es Bluetooth o "diente azul", que se hizo famoso por sus habilidades comunicativas uniendo Dinamarca y Noruega pero sobre todo por iniciar el proceso de cristianización de la sociedad vikinga.

En 1994 Ericsson inició un estudio para investigar la viabilidad de una interfase vía radio, de bajo costo y bajo consumo, para la interconexión

entre teléfonos móviles y otros accesorios con la intención de eliminar cables entre aparatos. El estudio partía de un largo proyecto que investigaba sobre unos multi - comunicadores conectados a una red celular, hasta que se llegó a un enlace de radio de corto alcance, llamado "MC link". Conforme éste proyecto avanzaba se fue descubriendo que éste tipo de enlace podía ser utilizado ampliamente en un gran número de aplicaciones, ya que tenia como principal virtud el que se basaba en un chip de radio relativamente económico.

A comienzos de 1997, según avanzaba el proyecto MC link, Ericsson fue despertando el interés de otros fabricantes de equipos portátiles. En seguida se vió claramente que para que el sistema tuviera éxito, un gran número de equipos deberían estar equipados con ésta tecnología. Esto fue lo que originó a principios de 1998, la creación de un grupo de interés especial (SIG), formado por 5 promotores que fueron: Ericsson ®, Nokia ®, IBM ®, Toshiba ® e Intel ®. La idea era lograr un conjunto adecuado de áreas de negocio, dos líderes del mercado de las telecomunicaciones, dos líderes del mercado de los PCS portátiles y un líder de la fabricación de chips. El propósito principal del consorcio fue y es, el establecer un estándar para la interfase aérea junto con su software de control, con el fin de asegurar la interoperatibilidad de los equipos entre los diversos fabricantes.

2.1.2 ARQUITECTURA

En lo referente a la arquitectura, la podemos dividir en 2 aspectos importantes, de acuerdo a todo lo que involucra y concierne al sistema inalámbrico Bluetooth:

- Núcleo del sistema (Core System) y
- Transporte de Datos

El núcleo del sistema es la arquitectura básica de todo dispositivo Bluetooth, sin incluir funciones de diversos protocolos tales como el SDP. Existen dos subsistemas dentro de la arquitectura: el controlador Bluetooth y el servidor Bluetooth. Generalmente se suele implementar una capa para la comunicación entre el controlador y el servidor, comúnmente llamado HCI.

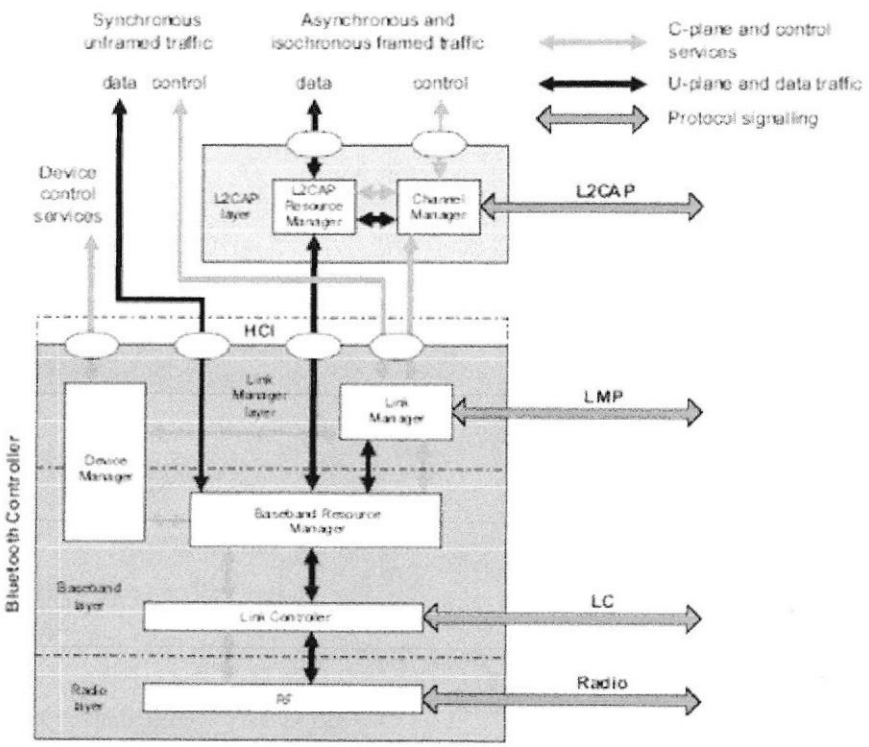


Figura 2.1: *Arquitectura del núcleo del sistema.*

El bloque Administrador del Canal (Channel Manager) es el responsable de crear, administrar y liberar canales L2CAP para el transporte de protocolos de servicios y aplicaciones. Este bloque utiliza el protocolo L2CAP para interactuar con el bloque similar de un dispositivo remoto (otro dispositivo Bluetooth) para crear estos canales L2CAP y conectarlos con sus apropiadas entidades.

El bloque Administrador de Recursos L2CAP (L2CAP Resource Manager) es el responsable de la administración y el orden de presentación de los fragmentos PDU de la capa banda base y relacionados, que aseguren que a los canales L2CAP comprometidos con la calidad de servicio (QoS) se les permita acceder a canales físicos. Se toma en cuenta que no existen contenedores de memoria ilimitados, ni ancho de banda infinito.

El bloque Administrador de Dispositivo (Device Manager) se encuentra en la capa banda base y controla en general el comportamiento de los dispositivos Bluetooth. Es el responsable de todas las operaciones del sistema que no están directamente relacionadas con el transporte de datos. Todo esto lo logra pidiendo el medio de transporte del Administrador de Recursos de Banda Base.

El bloque Administrador del Enlace (Link Manager) es responsable de la creación, modificación y liberación de enlaces lógicos, así como la actualización de parámetros relacionados con enlaces físicos entre

dispositivos, esto lo logra comunicándose con el administrador del enlace del dispositivo remoto usando el LMP, que es el Protocolo de Administración del Enlace.

El bloque Administrador de recursos de Banda Base (Baseband Resource Manager) es responsable de todos los accesos de radio, tiene dos funciones principales, el organizar el tiempo que poseen todas las entidades para negociar un acceso, y la segunda es negociar el acceso con estas entidades. Cuando los canales físicos no son alineados por sus time slots, este bloque se encarga de alinearlos entre los canales físicos existentes y los nuevos.

El bloque Controlador del enlace (Link Controller) es el responsable de codificar y decodificar los paquetes Bluetooth de carga útil y parámetros relacionados con los canales físicos, de transporte lógico y enlaces lógicos.

El bloque RF es responsable de la transmisión y recepción de paquetes de información sobre un canal físico. Un camino de control entre la capa banda base y RF permite que la capa banda base controle la sincronización y la frecuencia de la portadora del bloque RF.

En lo que se refiere a la arquitectura del transporte de datos podemos señalar tres capas principales que son: la capa física (Physical Layer)

donde tenemos los enlaces físicos y los canales físicos, la capa lógica (Logical Layer) donde tenemos los enlaces lógicos y los transportes lógicos y la capa L2CAP (L2CAP Layer) donde tenemos los canales L2CAP.

Es importante definir los tipos de tráfico: entramado, no entramado, confiable y no confiable. Los enlaces lógicos están asociados con el tipo de transporte lógico que indica lo que transportan. Los enlaces "C" para enlaces de control llevando mensajes LMP, los enlaces "U" para enlaces L2CAP llevando datos de usuario y los enlaces "S" por enlaces de ráfagas llevados por datos síncronos o isosíncronos sin formato.

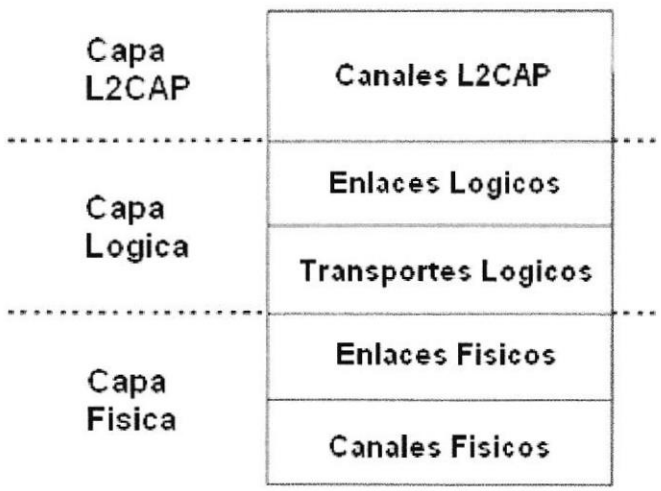


Figura 2.2: *Arquitectura del Transporte de Datos*

Dos dispositivos Bluetooth utilizan un canal físico compartido para comunicarse. Sus respectivos transceivers deben estar sincronizados en tiempo y frecuencia. Se definen 4 canales físicos, cada uno con un propósito diferente: Canal Piconet Básica y Canal Piconet Adaptada, ambos utilizados para la conexión entre dispositivos y son asociados a

una piconet específica, Canal Inquiry Scan, utilizado para el descubrimiento de dispositivos Bluetooth y el Canal Page Scan, utilizado para conectar dispositivos Bluetooth.

Los enlaces físicos se los maneja como un concepto virtual que no tiene representación directa en la estructura de un paquete de datos transmitido. Un enlace físico está siempre asociado con exactamente un canal físico. Los canales piconet básica y adaptada soportan un enlace físico, el cual puede estar "activo" o "estacionado". Este enlace está siempre presente cuando el esclavo se encuentra sincronizado en la piconet.

Un enlace físico entre el maestro y un esclavo está activo si el transporte lógico ACL existe entre los dispositivos. Las transmisiones desde el maestro pueden ser sobre sólo un enlace físico activo (a un esclavo específico) o sobre varios enlaces físicos (a un grupo de esclavos en una piconet). Un enlace físico entre el maestro y un esclavo está estacionado cuando el esclavo se mantiene sincronizado en la piconet, pero el transporte lógico ACL no está presente. Cuando un enlace activo es reemplazado por uno estacionado el transporte lógico ACL es removido implícitamente.

Existen cinco transportes lógicos que son:

- Asíncrono Orientado a la conexión (ACL): usado para llevar la señalización LMP y L2CAP. Cada esclavo activo en una piconet tiene un transporte lógico ACL con el maestro de la piconet. Es creado entre el maestro y un esclavo cuando el dispositivo se une a la piconet.
- Síncrono Orientado a la conexión (SCO): es un canal simétrico, punto a punto entre el maestro y un esclavo específico. Reserva slots en el canal físico y por lo tanto puede ser considerado como una conexión de conmutación de circuitos.
- Síncrono Orientado a la conexión Extendido (eSCO): la diferencia con el SCO es que el enlace eSCO ofrece retransmisión limitada de paquetes. Estas retransmisiones ocurren en los slots siguientes a los que estaban reservados.
- Difusión de esclavo activo (ASB): Usado para transportar el tráfico de usuario L2CAP hacia todos los dispositivos en la piconet que está conectada al canal físico que usa dicho ASB. No hay protocolo de reconocimiento y el tráfico es unidireccional desde el maestro hacia los esclavos.
- Difusión de esclavo estacionado (PSB): Usado para comunicaciones entre el maestro y los esclavos en estado estacionado. Es más complejo que los otros transportes lógicos ya que consiste en varias fases, cada una con un propósito diferente.

Existen tres tipos de enlaces lógicos:

- Enlace lógico de control ACL (ACL - C): usado para llevar la señalización LMP entre los dispositivos en la piconet. El enlace de control sólo se encuentra en el transporte lógico ACL y PSB. Si el ALC - C y el ACL - U están en el mismo transporte lógico, el ACL - C siempre tendrá prioridad sobre el otro.
- Enlace lógico de Usuario ACL (ACL - U): usado para llevar todos los frames isosincrónicos y asincrónicos. El enlace ACL - U se encuentra en todos los trasportes lógicos a excepción del SCO y eSCO.
- Enlace lógico de Usuario SCO/eSCO (SCO - S / eSCO - S): son usados como soporte para la entrega isocrónica de datos en una ráfaga sin trama. Estos enlaces están asociados a un transporte lógico, donde los datos son entregados en tamaños constantes a una tasa constante.

2.1.3 PROTOCOLOS

La especificación Bluetooth pretende que todas las aplicaciones sean capaces de operar entre sí. Para conseguir esta interoperabilidad, las aplicaciones en dispositivos remotos deben ejecutarse sobre una pila de protocolos idénticos. Para comunicarse con otros dispositivos Bluetooth, se requiere un hardware específico para Bluetooth, que incluye un módulo de banda base, así como otro módulo de radio y una antena.

Figura 2.3: *Bluetooth Hardware*

Además deberá existir un software encargado de controlar la conexión entre dos dispositivos Bluetooth; este software (Administrador del enlace) por lo general correrá en un microprocesador dedicado. Los administradores del enlace de diferentes dispositivos Bluetooth se comunicarán mediante el protocolo LMP.

Además habrá otros módulos de software, que constituirán la pila de protocolos, y garantizarán la interoperabilidad entre aplicaciones alojadas en diferentes dispositivos Bluetooth. La pila completa se compone tanto de protocolos específicos de Bluetooth (LM y L2CAP por ejemplo) como de protocolos no específicos de Bluetooth como son OBEX, UDP, TCP, IP, etc.

El objetivo principal a la hora de diseñar la torre de protocolos ha sido maximizar el número de protocolos existentes que se puedan reutilizar en las capas más altas para diferentes propósitos. A parte de todos estos protocolos, la especificación define el HCI, que se encarga de

proporcionar una interfaz de comandos al controlador de banda base, al gestor de enlace y nos da acceso al estado del hardware y a los registros de control.

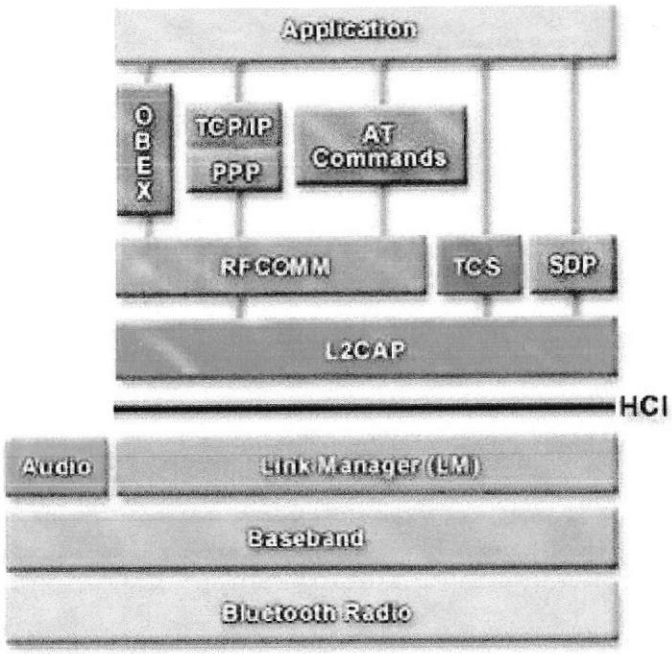


Figura 2.4: *Pila de Protocolos*

La capa de comunicación más baja es llamada Banda Base. Esta capa implementa el canal físico real. Emplea una secuencia aleatoria de saltos a través de 79 frecuencias de radio diferentes. Los paquetes son enviados sobre el canal físico, donde cada uno es enviado en una frecuencia de salto diferente. La capa de banda base controla la sincronización de las unidades Bluetooth y la secuencia de saltos en frecuencia, además es la responsable de la información para el control de enlace a bajo nivel como el reconocimiento, control de flujo y caracterización de la carga útil.

El Administrador del Enlace es el sistema que consigue establecer la conexión entre dispositivos. Se encarga del establecimiento, la autentificación y la configuración del enlace. El Administrador del Enlace localiza a otros gestores y se comunica con ellos gracias al protocolo de gestión del enlace LMP. Para poder realizar su función de proveedor de servicio, el administrador del enlace utiliza los servicios incluidos en el controlador de enlace. El LMP básicamente consiste en un número de PDUs que son enviados de un dispositivo a otro.

El HCI proporciona una interfaz de comandos para el controlador de banda base y para el gestor de enlace, y permite acceder al estado del hardware y a los registros de control. Esta interfaz proporciona una capa de acceso homogénea para todos los dispositivos Bluetooth de banda base. El hardware de la capa HCI intercambia comandos y datos con el firmware del HCI presente en el dispositivo Bluetooth. El controlador del bus físico proporciona a ambas capas de HCI la posibilidad de intercambiar información entre ellas. Una de las tareas más importantes de HCI que se deben realizar es el descubrimiento automático de otros dispositivos Bluetooth que se encuentren dentro del radio de cobertura. Esta operación se denomina Inquiry.

El protocolo L2CAP proporciona servicios de datos tanto orientados a conexión como no orientados a conexión a los protocolos de las capas superiores, junto con facilidades de multiplexación y de segmentación y reensamblaje. L2CAP permite que los protocolos de capas

superiores puedan transmitir y recibir paquetes de datos L2CAP de hasta 64 KBs de longitud. L2CAP se basa en el concepto de canales. Un canal es una conexión lógica que se sitúa sobre la conexión de banda base. Cada canal se asocia a un único protocolo. Cada paquete L2CAP que se recibe a un canal se redirige al protocolo superior correspondiente. Varios canales pueden operar sobre la misma conexión de banda base, pero un canal no puede tener asociados más de un protocolo de alto nivel.

El protocolo RFCOMM proporciona emulación de puertos seriales a través del protocolo L2CAP. Este protocolo se basa en el estándar de la ETSI denominado TS 07.10. RFCOMM es un protocolo de transporte sencillo, con soporte para hasta 9 puertos seriales RS - 232 (EIATIA - 232 - E). El protocolo RFCOMM permite hasta 60 conexiones simultáneas (canales RFCOMM) entre dos dispositivos Bluetooth. Para los propósitos de RFCOMM, un camino de comunicación involucra siempre a dos aplicaciones que se ejecutan en dos dispositivos distintos (los extremos de la comunicación). Entre ellos existe un segmento que los comunica. RFCOMM pretende cubrir aquellas aplicaciones que utilizan los puertos serie de las máquinas donde se ejecutan. El segmento de comunicación es un enlace Bluetooth desde un dispositivo al otro (conexión directa). RFCOMM trata únicamente con la conexión de dispositivos directamente, y también con conexiones entre el dispositivo y el modem para realizar conexiones de red. RFCOMM puede soportar otras configuraciones, tales como

módulos que se comunican vía Bluetooth por un lado y que proporcionan una interfaz de red cableada por el otro.

El SDP permite a las aplicaciones cliente descubrir la existencia de diversos servicios proporcionados por uno o varios servidores de aplicaciones, junto con los atributos y propiedades de los servicios que se ofrecen. Estos atributos de servicio incluyen el tipo o clase de servicio ofrecido y el mecanismo o la información necesaria para utilizar dichos servicios. SDP se basa en una determinada comunicación entre un servidor SDP y un cliente SDP. SDP proporciona un mecanismo para el descubrimiento de servicios y sus atributos asociados, pero no proporciona ningún mecanismo ni protocolo para utilizar dichos servicios.

El TCS binario es un protocolo que define la señalización de control de llamadas, para el establecimiento y liberación de una conversación o una llamada de datos entre unidades Bluetooth. Además, éste ofrece funcionalidad para intercambiar información de señalización no relacionada con el progreso de llamadas.

La capa de Audio es una capa especial, usada sólo para enviar audio sobre Bluetooth. Las transmisiones de audio pueden ser ejecutadas entre una o más unidades usando muchos modelos diferentes. Los datos de audio no pasan a través de la capa L2CAP, pero sí

directamente entre los dos dispositivos Bluetooth después de abrir un enlace.

2.1.4 PERFILES

Los perfiles han sido desarrollados con el fin de describir como se crean los modelos de usuario a nivel de aplicaciones. Los modelos de usuario describen un número de escenarios donde exista transmisión radio vía Bluetooth. Un perfil puede ser descrito como una capa vertical sobre la pila de protocolos, en otras palabras, define que protocolos son obligatorios para el funcionamiento del perfil en particular. Los perfiles se usan para aminorar la cantidad de posibles problemas de interoperabilidad entre dispositivos de diferentes fabricantes.

Las especificaciones de un perfil dado contienen información de los siguientes temas:

- Dependencias con otros perfiles.
- Formatos sugeridos sobre interfaz de usuario.
- Partes específicas de la pila de protocolos usadas por el perfil.

En el apéndice D se definen los perfiles que utiliza Bluetooth hasta el momento.

2.1.5 PROCEDIMIENTOS DE CONEXIÓN

Normalmente la conexión entre dos dispositivos Bluetooth ocurre de la siguiente manera: si nada es conocido entre los dispositivos, se debe comenzar con los procedimientos de Inquiry y Paging. Si en cambio ya se conocen algunos detalles entre los dispositivos, sólo se necesita el procedimiento de Paging.

El procedimiento de Inquiry permite a un dispositivo ser descubierto por otros dispositivos que se encuentren en su rango y a su vez determina las direcciones y sus respectivos relojes. Se involucra un dispositivo (la fuente) el cual envía paquetes de inquirí, estado "inquiry" y después recibe la respuesta correspondiente. El dispositivo que recibe los paquetes de inquiry (el destino) debe encontrarse en el sub - estado de "inquiry scan" para poder recibir dichos paquetes. Después el dispositivo destino entrará al sub - estado de "inquiry response" y enviará una respuesta de inquiry a la fuente. Luego que el procedimiento de inquiry se ha completado, se establece la conexión mediante el procedimiento de Paging.

El procedimiento de Paging se lo utiliza para establecer la conexión entre dos dispositivos Bluetooth. Sólo se requiere la dirección del dispositivo para establecer la conexión. El conocimiento del reloj aceleraría este procedimiento. El dispositivo que establece la conexión llevará el procedimiento de Paging y automáticamente se convertirá en el maestro de la conexión. Un dispositivo (la fuente) alerta a otro dispositivo (el destino). El destino recibe la alerta. El destino envía una

respuesta a la fuente. La fuente envía un paquete FHS al destino. El destino envía una segunda respuesta a la fuente. El destino y la fuente se cambian a los parámetros del canal de la fuente. La conexión empieza con el envío de un paquete POLL por el maestro para verificar si el esclavo se ha cambiado a las condiciones del canal (reloj y FH). El esclavo puede responder con cualquier tipo de paquete.

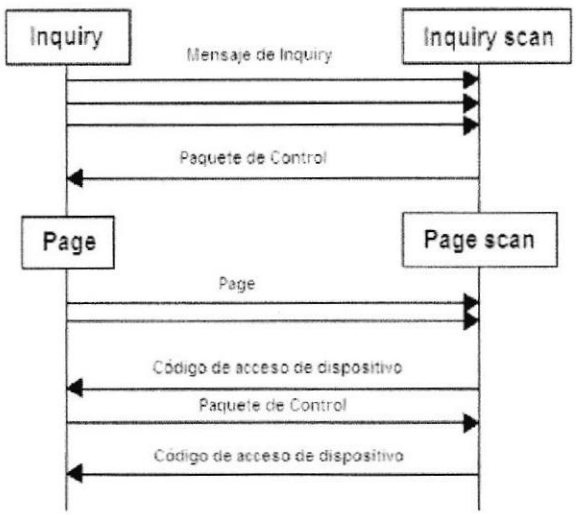


Figura 2.5: *Establecimiento de la conexión.*

2.2 REDES AD - HOC

Las redes Ad - Hoc son el nuevo paradigma de las comunicaciones inalámbricas entre hosts móviles (a los cuales se llamaran nodos), este tipo de redes también conocidas como MANETs, son redes inalámbricas que no requieren ningún tipo de infraestructura fija ni administración centralizada, donde las estaciones, además de ofrecer funcionalidades de estación final, deben proporcionar también servicios de encaminamiento, retransmitiendo paquetes entre aquellas estaciones que no tienen conexión inalámbrica directa.

Las redes Ad - Hoc se hicieron populares cuando el estándar IEEE 802.11 empezó a investigar el comportamiento de este tipo en laptops y dispositivos móviles, es así que en mediados de los noventas se obtuvieron gran cantidad de papers para evaluar los diferentes tipos de protocolos y habilidades asumiendo la movilidad, el rango de cobertura, la relación de los paquetes y las tasas de transferencia.

2.2.1 FUNDAMENTOS

Estas redes requieren de nuevos algoritmos, protocolos y "middleware" (algún controlador que ayude con la gestión de paquetes, transmisiones, etc.). Los protocolos de encaminamiento desarrollados para redes cableadas no se adaptan al entorno altamente dinámico de las redes Ad - Hoc.

El interés que las redes Ad - Hoc han despertado dentro de "La Fuerza de Tareas de Ingeniería de Internet" (IETF) llevo a la formación de un nuevo grupo de trabajo denominado "Redes Móviles Ad – Hoc" (MANET), cuyo principal objetivo es el de estimular la investigación en el área de las redes Ad - Hoc.

Existen una gran variedad de algoritmos de enrutamiento para este tipo de redes, entre los más utilizados tenemos:

- AODV
- DSR

- TORA
- OLSR
- DSDV

Tanto AODV como DSR utilizan un encaminamiento reactivo o bajo demanda, en el cual las rutas a utilizar para un determinado destino sólo se calculan cuando éstas son necesarias. Estos protocolos, intentan reducir así la sobrecarga generada por los mensajes de actualización periódicos. TORA a excepción de los otros dos, no usa el método del camino mas corto para llegar a otro nodo, este usa los pesos de tal manera que dos nodos no tengan el mismo peso.

El principal inconveniente de todos los protocolos bajo demanda es el retardo inicial que introducen y que puede representar una seria limitación en aplicaciones que requieran asegurar una determinada calidad de servicio, como lo seria el audio y video.

Por su parte OLSR utiliza un encaminamiento proactivo en donde todas las rutas a todos los posibles destinos se calculan a priori y además, estas se mantienen actualizadas en todo momento utilizando para ello mensajes de actualización periódicos.

Al igual que OLSR, DSDV utiliza un encaminamiento proactivo el cual se basa en poner a los nodos números secuenciales, de tal manera

que las actualizaciones se hacen según el número que se asigne o se tenga que asignar en caso de que se incrementen los nodos en la red.

2.2.2 CONFIGURACIÓN DE REDES BLUETOOTH

En la actualidad Bluetooth es un estándar que es ampliamente utilizado en dispositivos móviles tal como lo son laptops, teléfonos Celulares, PDAs, audífonos, Dispositivos de Interfaz Humana (Mouse, teclados, gamepads, etc.), y en muchas aplicaciones, tanto es así, que para configurar una red Bluetooth, primero deberíamos de saber que aplicación es la que vamos a usar. Básicamente las redes Bluetooth son redes tipo Ad - Hoc, con topología tipo estrella por lo general, donde se obtiene comunicación punto a punto o punto a multipunto.

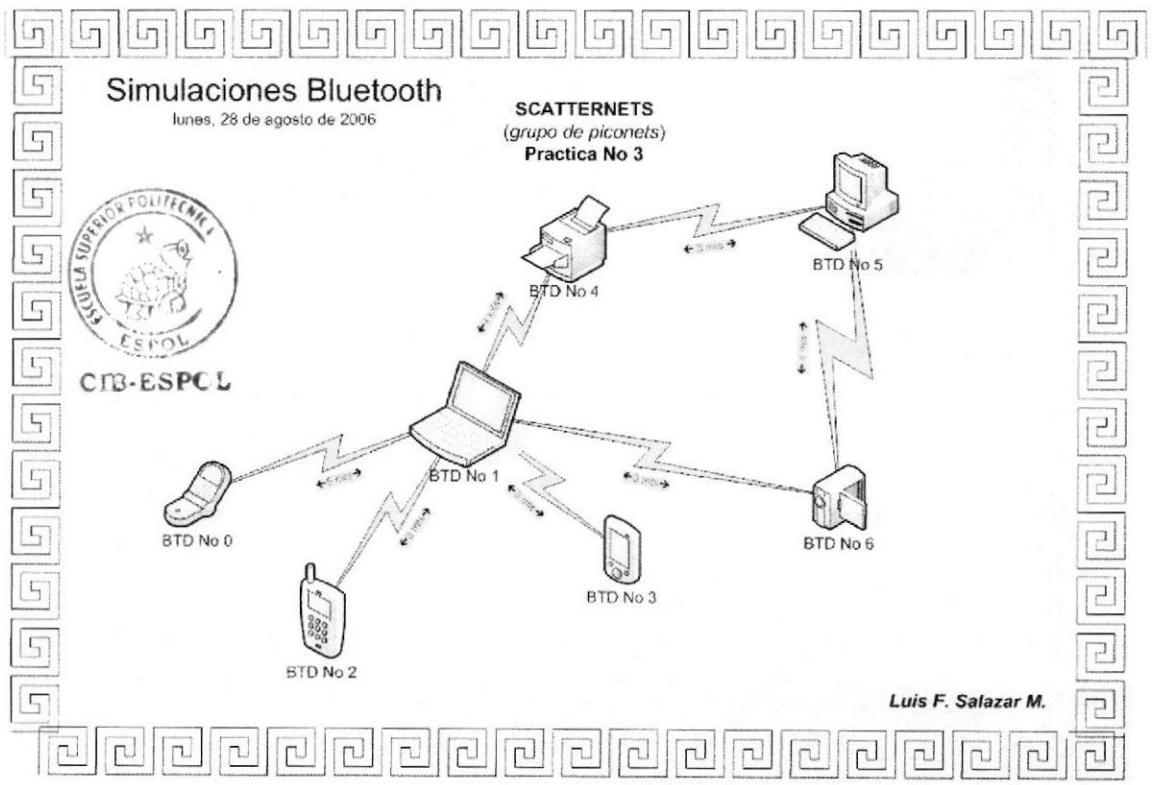


Figura 2.6: *Red Ad - Hoc Bluetooth.*

Los típicos modos operacionales de los dispositivos Bluetooth corresponden a que estén conectados a otros nodos intercambiando datos, por ejemplo en una piconet. Entre los principales modos tenemos:

- Modo Activo: El dispositivo participa activamente en el canal. El maestro programa la transmisión basado en la demanda del tráfico hacia y desde los diferentes esclavos. Además soporta transmisiones regulares para mantener a los esclavos sincronizados al canal. Si un esclavo activo no es requerido en la transmisión, puede "dormir" hasta la siguiente transmisión del maestro.

- Modo Sniff: Dispositivos sincronizados a una piconet pueden entrar a modos de ahorro de energía en el cual la actividad del dispositivo disminuye. En este modo, un esclavo está conectado a la piconet a una tasa reducida, disminuyendo su ciclo de trabajo.

- Modo En Espera: El maestro puede poner a los esclavos en este modo en el cual sólo el reloj interno se encuentra trabajando. Los esclavos también pueden pedir cambiar su estado a modo en espera. La transmisión de datos re - comienza instantáneamente cuando los dispositivos dejan el modo en espera.

- Modo Estacionado: El dispositivo aún se encuentra sincronizado con la piconet pero no participa en el tráfico de datos. Los dispositivos

en este modo han dado su dirección MAC y ocasionalmente escuchan el tráfico del maestro para re - sincronizarse y revisar los mensajes broadcast.

CAPÍTULO 3:

3. DESCRIPCION DEL PROYECTO

3.1 SOLUCION PLANTEADA

Recordando el problema, en este caso la necesidad de poseer una herramienta de análisis y simulación de redes Bluetooth de una manera didáctica para la facultad, lo primero que necesitamos es un simulador que tenga las siguientes cualidades:

- Flexible
- Poco costo
- Confiable
- Que esté orientado a lo que son comunicaciones móviles
- Que use lenguajes conocidos o no muy complicados
- Que posea documentación básica del mismo
- Que sea utilizado a nivel mundial

Debido a estas razones el simulador escogido fue el NS2. Para poder desarrollar simulaciones de redes Bluetooth, éste debe contar con una librería desarrollada para tal fin.

En lo que respecta a las simulaciones de redes Bluetooth necesitamos ciertas condiciones haciendo referencia al simulador, en este caso NS2, por lo que debemos tener en cuenta lo siguiente:

- Esta librería sea creada para NS2
- La librería debe reflejar los procesos definidos por el estándar Bluetooth
- Que presente y pueda dar resultados fehacientes en problemas relacionados a Bluetooth
- Que tenga un buen nivel de desarrollo
- Tener una comunicación con los desarrolladores de la librería para poder establecer nuestros objetivos dentro de la misma
- Compatibilidad gráfica (opcional)

Decidimos optar por UCBT ya que se encuentra en desarrollo por lo que podemos libremente crecer en conjunto con la librería. Nuestro aporte será el de agregar mayor información para la interpretación de los resultados e implementar funciones las cuales no posee esta librería para ser apreciadas y utilizadas con la herramienta gráfica NAM. Básicamente la lista de correos, habilitado por el creador de la librería, es una fuente de conocimientos esparcidos donde podemos obtener información acerca de la librería, como funciona y demás detalles de las simulaciones de redes Bluetooth del cual nos valdremos para poder realizar nuestra contribución a la librería.

Para solucionar los principales problemas que presenta la librería UCBT, entre los cuales tenemos que actualmente carece de una interfaz gráfica

que pueda ser manipulada por el usuario además de no ofrecer ningún detalle de la simulación, decidimos incorporar a la librería, en algunas funciones pertenecientes a esta, el código necesario que nos permita realizar las principales acciones tales como el posicionamiento y el movimiento de los nodos para que se las puedan apreciar durante la simulación en el NAM.

Además para revisar el estado de los nodos en un instante de tiempo tenemos que recurrir al archivo de salida del simulador, en otras palabras, éste no ofrece información del tiempo en que ocurre el cambio de estado de los nodos y cuales son estos.

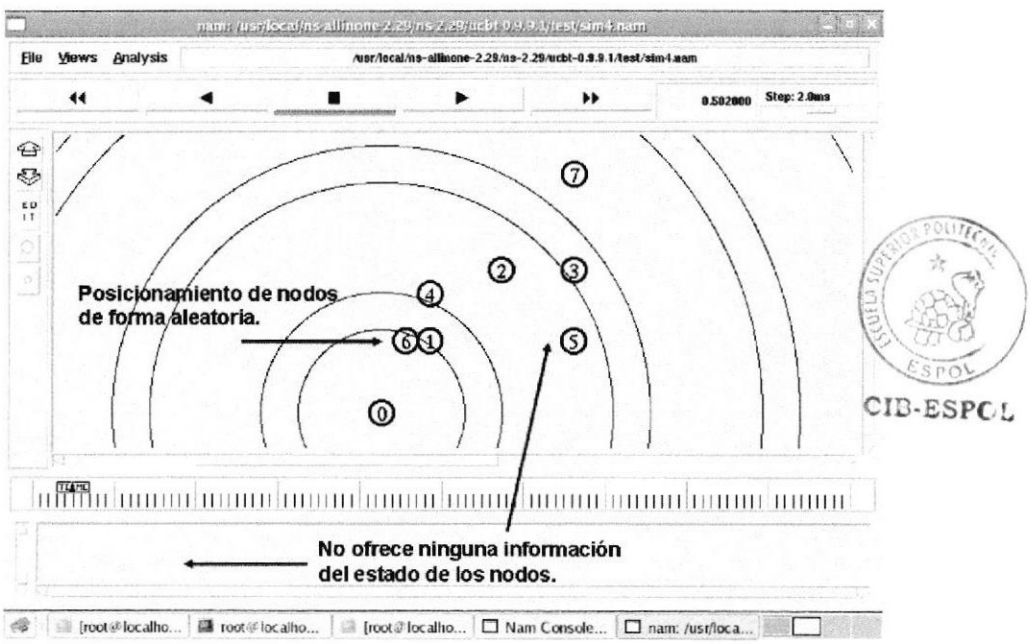


Figura. 3.1*Interfaz gráfica de la librería UCBT, antes de las modificaciones.*

Lo que planteamos como solución es modificar las funciones necesarias que nos permitan realizar las acciones requeridas para desarrollar el ambiente gráfico, dado que la versión de la librería UCBT que utilizamos no ofrece tales opciones. Como podemos observar en la Figura 3.1 el

posicionamiento de los nodos es de forma fija para todas las simulaciones que se realicen. Por otro lado no se presenta ninguna información acerca del estado de los nodos, haciendo la interfaz gráfica de poca utilidad para el aprendizaje. Solucionamos ese problema incorporando a la parte gráfica información variada durante la simulación, entre una de las más importantes tenemos el estado del nodo colocado sobre cada uno de ellos.

Como aportes adicionales mostramos en forma de comentario información del nodo como lo es la identificación del instante de tiempo en el cual éste cambia de un estado a otro. Además se le colocarán marcas amarillas a los nodos que alcanzan el estado de Maestro, marcas verdes a los nodos que alcanzan el estado de Esclavo y marcas rojas a los nodos móviles cuyo destino se encuentre fuera del área de cobertura de la piconet a la cual pertenecen con respecto al maestro, para obtener una mejor apreciación de los resultados en cada instante de tiempo mientras dure la simulación. Los resultados después de las modificaciones realizadas a la librería se pueden apreciar en la Figura. 3.2.

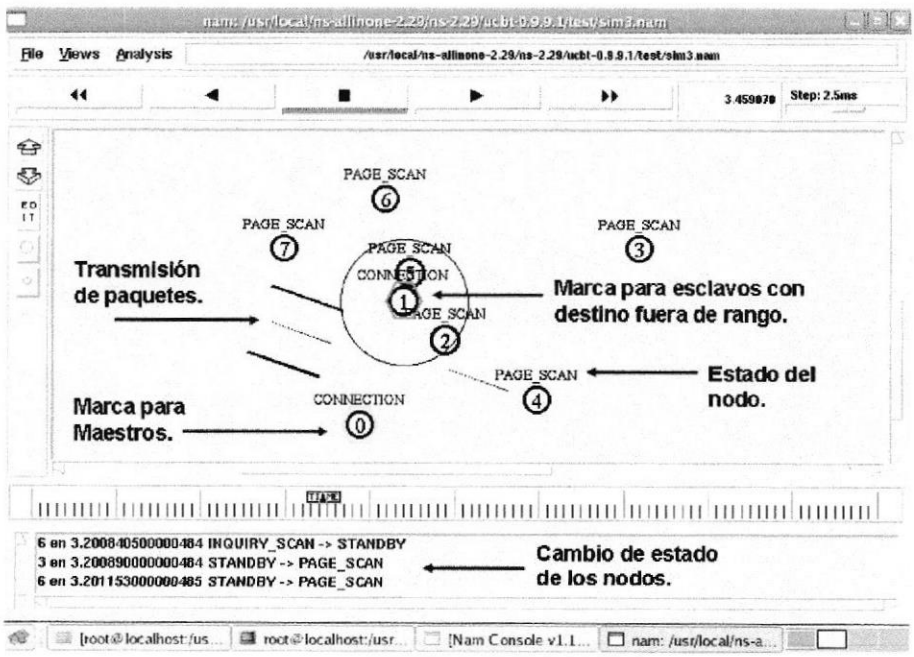


Figura. 3.2 *Interfaz gráfica de la librería UCBT, después de las modificaciones.*

3.2 DESCRIPCIÓN DEL SOFTWARE

3.2.1 DESCRIPCIÓN DE NS2

NS2 inició en 1989 con el desarrollo de Real Network Simulator por S. Keshav. Seis años después, en 1995, el Grupo de Investigación de Redes del Laboratorio Nacional Lawrence Berkeley (LBNL), basándose en el trabajo realizado por Keshav, lanzó el NS. Después en 1997 la Universidad de California en Berkeley (UCB) desarrolló NS2 el cual incorpora las funcionalidades de la versión anterior y se desarrolla continuamente manteniendo la compatibilidad con versiones anteriores.

NS2 es un simulador de eventos discretos dirigido hacia la investigación en el área de networking. NS2 provee un soporte sustancial para la simulación de TCP, ruteo y protocolos multicast sobre redes cableadas e inalámbricas (locales y satelitales).

Con NS2 podemos simular varios escenarios con los protocolos de transporte de paquetes UDP y TCP; protocolos de ruteo en redes cableadas como vector - distancia y estado - enlace; protocolos de ruteo de redes Ad - Hoc, como DSR, AODV, TORA; protocolos de MAC, 802.3, 802.11 (MAC inalámbrica), disciplinas de programación como DropTail, RED, DRR, etc.

Debido a la flexibilidad del NS2 se pueden implementar los arreglos que se requieran, usando los protocolos que están incorporados al simulador o inclusive creando nuevos protocolos con código total o parcialmente nuevo.

NS2 utiliza dos lenguajes de programación para dos tareas especificas, una de ellas se usa en el manejo de protocolos y paquetes a nivel de bytes, y se lo implementa usando C++, y la otra permite el control y la modificación dinámica de eventos, para ello usamos el lenguaje Otcl. Teniendo en cuenta lo siguiente, que C++ se demora más para establecer los cambios dinámicamente, pero Otcl a su vez logra manejar efectivamente estos cambios, siendo C++ el que se encarga de tiempos de ejecución rápidos. Ambos lenguajes trabajan simultáneamente como se puede ver en la Figura 3.3.

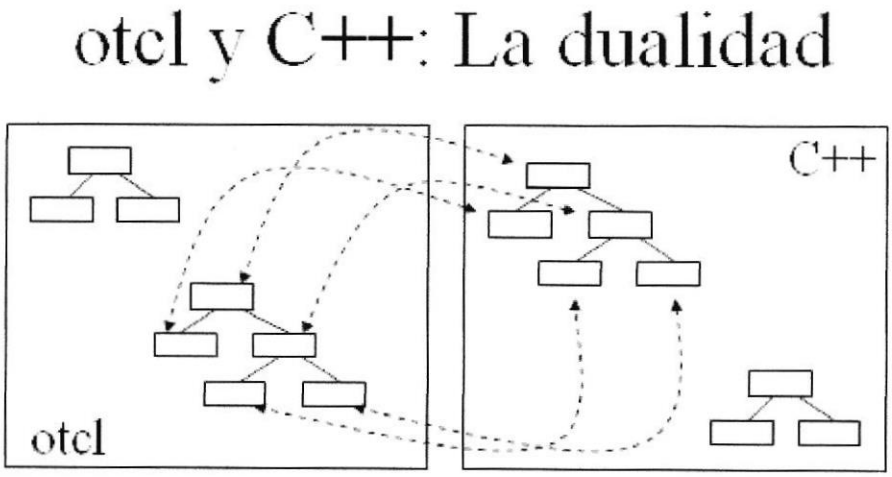


Figura 3.3: *lenguajes que usa el NS2*

Ambos lenguajes deben trabajar simultáneamente, por lo que se utiliza un intérprete el cual se encarga de la gestión de estos lenguajes, para lo cual NS2 utiliza el tclcl.

Los escenarios de simulación se realizan en archivos llamados scripts escritos en lenguaje tcl, con lo cual se obtienen archivos de trazas estos a su vez pueden generar varios resultados para usarlos en el NAM. Con estos archivos se realiza la visualización y análisis de las simulaciones.

3.2.2 DESCRIPCIÓN DE LA LIBRERÍA UCBT

UCBT (Universidad de Cincinnati - Bluetooth) es un módulo de red Bluetooth basado en NS2, el cual simula las operaciones de las redes Bluetooth en gran detalle. La mayoría de las especificaciones en banda base y capas superiores como LMP, L2CAP, BNEP han sido simuladas en UCBT, incluyendo el esquema de salto de frecuencia (frequency hopping), descubrimiento de dispositivos, establecimiento de la conexión, estados de parqueo, en espera, intercambio de roles, negociación de paquetes multi - slot, conexión de voz SCO, etc.

Hay una provisión para constituir un grupo de dispositivos Bluetooth y su formación con hasta 8 dispositivos Bluetooth que es conocido como una piconet. Este también permite un número de piconets a ser conectadas entre ellas, usando nodos puente, en una estructura más grande conocida como scatternet.

UCBT no es el primer simulador Bluetooth basado en NS2. Bluehoc, desarrollado por IBM y su extensión para Scatternets llamado Blueware por el MIT, ambos son predecesores de UCBT, sin embargo con más de 28,000 líneas de código en C++, UCBT es probablemente el más certero simulador de redes Bluetooth. UCBT se adapta al perfil PAN con el protocolo BNEP.

UCBT deriva el reloj interno en un conteo, el cual es muy importante para la simulación de la sincronización y certeza de los protocolos de organización. UCBT también incluye el reciente EDR de la especificación para poder simular nuevos dispositivos con tasas de 2 o hasta 3 Mbps. Una de nuestras principales contribuciones es que UCBT provee un trabajo de tramas flexibles que permite conducir a la investigación del comportamiento de scatternets.

Una scatternet requiere de tiempo compartido de algunos dispositivos (puentes) entre las piconets. Coordinar la presencia de ordenamiento de los nodos puentes dentro de una gran mezcla de dispositivos como lo es una scatternet, es bastante complicado. UCBT provee algunos algoritmos para la organización de múltiples nodos puente, habilitando así la operación de las scatternets. Scatternets prototipo auto-organizadas han sido diseñadas y simuladas.

Nuestro trabajo se centra principalmente en la capa banda base del modulo UCBT, ya que aquí se genera importante información que nos sirvió para poder analizar en que momento y bajo que circunstancias realizábamos las acciones que son nuestro aporte a este módulo.

Esta capa realiza la simulación de los procesos de inquiry, inquiry scan, page, page scan entre otros, además suministra información, en algunas de sus funciones, acerca del estado de los nodos, los tiempos de los cambios de estado y características del envío de paquetes, así como los nodos que intervienen en este proceso. También nos permite complementar funciones necesarias para poder obtener salidas en el NAM, tal es el caso de la función para producir el movimiento de los nodos incluida en esta librería.

Además aprovechando el hecho de que la simulación de esta capa genera constantemente información de los estados de los nodos y la información del envío de paquetes, podemos imprimir en la salida del NAM información que puede ayudar a comprender las acciones generadas durante la simulación como los son la formación de las piconets o scatternets.

3.2.3 DESCRIPCIÓN DE LAS MODIFICACIONES A LA LIBRERÍA UCBT

Para realizar las modificaciones a la librería y obtener los resultados deseados usaremos dos herramientas, estas son:

1. La función ***tcl.eval (char* x)*** **[*]** de la clase "Tcl instante" que esta declarada en ***~tclcl/Tcl.cc***, para obtener acceso a esta instancia debemos declarar la siguiente línea en cada función de la librería donde se use ***tcl.eval (char* x)***.

Tcl & tcl = Tcl :: instance ();

2. El formato de las trazas del archivo del NAM, para cada evento **[*]** .

Utilizaremos estas herramientas en ciertas funciones que se encuentran dentro de los archivos ***baseband.cc, wnode.cc y ns-btnode.tcl*** estas funciones fueron seleccionadas por ofrecernos información acerca del estado, los posicionamientos y los tiempos en que ocurren los cambios de estados en cada nodo así como también la identificación del nodo que en un tiempo determinado esta siendo tratado.

La mayor cantidad de código se añadió en el archivo ***baseband.cc***, aquí además del código añadido se declararon dos variable globales, estas son ***Coor [50] [8]*** de tipo doble y ***sec*** de tipo entero, el funcionamiento de cada una de estas variables de explicaran mas adelante.

La variable ***Coor [50] [8]*** es una matriz que acepta un número máximo de 50 nodos por defecto y almacena importante información como lo

son las coordenadas de los nodos y demás banderas utilizadas para identificar ciertos estados de los nodos como lo son si es maestro, el nodo esta marcado, si el nodo esta conectado, etc. Esta información servirá para poder realizar acciones dependiendo del caso en las funciones que se han modificado dentro del archivo ***baseband.cc,*** en la Tabla 3.1 se muestra la matriz definida por la variable ***Coor [50] [8]***.

	Pos X	Pos Y	f Coor	f Mark MST	f Mark OUTR	f es MST	f MSTid	f CONNET
0	0,00	0,00	1	0	0	1	0	0
1	5,00	2,36	0	0	1	0	0	1
2	10,50	25,36	0	1	0	1	0	1
3	1,00	0,00	0	1	1	0	3	1
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
47	25,69	14,82	1	0	0	0	6	1
48	12,36	1,65	0	0	1	1	9	1
49	13,54	2,54	1	1	1	0	5	1

Tabla. 3.1 *Matriz definida por la variable Coor [50] [8].*

Las filas de la matriz representa el identificador del nodo aprovechando el hecho que en las simulaciones se crean nodos de manera secuencial, de esta manera asociamos las filas de la matriz con el identificador de cada nodo es por eso que en la mayoría de los casos usaremos la variable ***Coor [n] [m]***, donde **n** tiene un rango [0....49] y **m** un rango [0....7], de la siguiente manera: ***Coor [bd_addr_] [m], Coor [sender] [m]*** y ***Coor [receiver] [m]***, siendo bd_addr_, sender, receiver las variables definidas por la librería UCBT en el archivo ***baseband.cc*** como identificadores de los dispositivos.

Los datos almacenados en las localidades de la columna ***Pos X*** (***Coor [n] [0]***) representan las posiciones en **X** de los nodos, si hacemos la analogía con las coordenadas (x, y), de igual forma los datos

almacenados en las localidades de la columna ***Pos Y*** (***Coor [n] [1]***) representan las posiciones en **Y** del nodo.

Los datos almacenados en la columna ***f Coor*** (***Coor [n] [2]***) son banderas que nos indican: si su valor es 0, el nodo no se ha movido y si su valor es 1, el nodo ya ha realizado por lo menos un movimiento. Usamos esta bandera debido a que la librería UCBT maneja una estructura llamada ***node_*** en la cual se encuentran definidas las variables ***X_*** (***node_->X_***), ***Y_*** (***node_->Y_***) y ***Z_*** (***node_->Z_***).

Estas variables guardan información de la posición de los nodos pero cuando se realiza el primer movimiento estos valores son distintos a la posición de destino del nodo, así que a partir del primer movimiento estos valores no son útiles como posición de partida para un nuevo movimiento del mismo. Si usamos estas variables en la simulación para otro movimiento se creará efectos de brincos al invocarse la función que realiza el efecto de desplazamiento del nodo.

Por ello el primer valor que almacenara ***Coor [n] [0]*** y ***Coor [n] [1]*** serán los valores que contengan las variables ***node_->X_*** y ***node_->Y_***, la variable ***node_->Z_*** es descartada porque no trabajamos en tres dimensiones, cuando se realiza el primer movimiento los datos almacenados en ***Coor [n] [0]*** y ***Coor [n] [1]*** serán la posición de destino del nodo, es por ello que antes de asignar el valor a estas variables verificamos el estado de ***f Coor*** (***Coor [n] [2]***).

Los datos almacenados en la columna ***f Mark MST*** (***Coor [n] [3]***) son banderas que nos indican si el nodo es maestro y es usada para ponerle una marca que lo distinga del resto de nodos, esta marca es un circulo de color amarillo colocado sobre el nodo, esto ocurre cuando el valor de esta bandera para el correspondiente nodo es 1. Cuando el nodo sale del estado de maestro entonces esta bandera toma el valor de 0 y la respectiva marca que lo identificaba como tal debe ser removida.

Los datos almacenados en la columna ***f Mark OUTR*** (***Coor [n] [4]***) son banderas usadas cuando realizamos algún movimiento de nodos, se activa cuando el nodo móvil realiza un movimiento cuyo destino se encuentra fuera del rango de la piconet a la que pertenece, caso contrario permanece en 0, esto nos es útil para colocarle una marca a ese nodo. Esta marca es un hexágono de color rojo colocado sobre el nodo cuando se ha activado esta bandera para poder diferenciarlo del resto, de igual forma si el mismo nodo realiza otro movimiento cuyo destino este dentro de rango de la piconet en la que se encuentra, la marca que este poseía será removida.

Los datos almacenados en la columna ***f es MST*** (***Coor [n] [5]***) son banderas usadas para indicarnos si el nodo es maestro en el caso de estar esta activa, con el valor de 1, y que no lo es cuando esta tiene el valor de 0. Los datos almacenados en la columna ***f MSTid*** (***Coor [n]***

[6]) en cambio nos indican la piconet a la cual pertenece cada nodo activo y lo hacemos asociando al nodo con el identificador del maestro al cual están conectados.

Los datos almacenados en la columna ***f CONNET*** (***Coor [n] [7]***) son banderas que nos indican si el nodo es un esclavo en caso de estar activa con el valor de 1 y que no lo es cuando tiene el valor de 0, su funcionamiento en el equivalente al de ***f es MST*** pero para el caso de esclavos.

Para realizar las acciones que se visualizarán en NAM hacemos uso de los datos en la matriz ***Coor [50] [7]*** mediante validaciones y el uso de la función ***tcl.eval (char* x)*** para hacer que se ejecute la correspondiente acción, con el uso del correspondiente formato de las trazas que se almacenan en el archivo de NAM.

Para el caso de colocar los estados de los nodos sobre cada uno de ellos debemos utilizar el siguiente formato:

tcl.evalf ("puts $namfile \"n -t %.15f -s %d -S DLABEL -l %s -L %s\"", clkoffset, bd_addr_, nuevoest, lastest)

Donde ***clkoffset*** representa el instante de tiempo en que aparecerá esta acción durante la simulación, ***bd_addr_*** es el identificador del

nodo, ***nuevoest*** es el nuevo estado del nodo y ***lastest*** el estado anterior.

Para colocar comentarios del instante en que el nodo cambia de estado y cuales son estos, primero crearemos una variable tipo cadena de caracteres, la cual llamaremos ***anotacion***, que contenga el comentario que aparecerá en la ventana de NAM de la siguiente manera:

sprintf (anotacion, "%d en %.15f %s -> %s", bd_addr_, Scheduler::instance().clock(), lastest, nuevoest)

Donde ***bd_addr_*** es el identificador del nodo, ***Scheduler::instance().clock()*** es el instante de tiempo en que ocurre el cambio de estado, ***sec*** es una variable global mencionada anteriormente cuya función es generar la secuencia en la que aparecerán los comentarios en la pantalla del NAM, ***lastest*** y ***nuevoest*** son el anterior estado y el estado nuevo. Luego para ejecutar esta acción en el simulador utilizaremos el siguiente formato:

tcl.evalf ("puts $namfile \"v -t %.15f -e sim_annotation %.15f %d %s\"", clkoffset, clkoffset, sec, anotacion)

Cuando tengamos que ponerle las marcas (Amarillas) a los nodos que alcanzan el estado de maestro utilizaremos el siguiente formato:

tcl.evalf ("puts $namfile \"m -t %.15f -s %d -n m%d -c yellow -h circle\"", clkoffset, bd_addr_, bd_addr_)

Y para remover estas marcas cuando estos nodos salgan del estado de maestros utilizaremos este formato:

tcl.evalf ("puts $namfile \"m -t %.15f -s %d -n m%d -c yellow -h circle -X\"", clkoffset, bd_addr_, bd_addr_)

La única diferencia entre los dos formatos es el parámetros ***–X [*]***, este formato le indica al interprete del NAM que si un nodo tiene una marca puesta se la remueva.

El segmento de código agregado para la parte anteriormente descrita, la cual se encuentra en la función ***change_state()***, es descrito en el algoritmo que se muestra en la siguiente figura :

```
change_state ()
Tcl & tcl = Tcl :: instance ();
anotacion [100], nuevoest [100], lastest [100]        //cadenas de strings
... ... ...
if (state_str()!="NEW_CONNECTION_SLAVE")) then    //es esclavo?
        f es SLV [bd_addr_] = 1
        //bd_addr_ es el identificador del nodo.
if ((state_str()!="NEW_CONNECTION_MASTER")        //es maestro?
        sprintf(nuevoest, "NEW_CONNECTION_MST")
        if (f Coor [bd_addr_] ! = 1) then    //primer movimiento?
                f Coor [bd_addr_] = 1                //marcar primer movimiento
                Pos X [bd_addr_] = node_->X_    //almaceno la posición en X
                Pos Y [bd_addr_] = node_->Y_    //almaceno la posición en Y
                f es MST = 1                //el nodo es como maestro
        if (f Mark MST [bd_addr_] ! = 1 ) then        //el nodo tiene alguna marca?
                f Mark MST [bd_addr_] = 1        //colocar marca al maestro
                ... ... ...
                tcl.evalf ("puts $namfile \"m -t %.15f -s %d -n m%d -c yellow -h hexagon\"",
                clkoffset, bd_addr_, bd_addr_)
                // instrucción para realizar la acción de colocar marca al maestro
        else
                sprintf (nuevoest, "%s", state_str())
```

```
                if (! fesMST [bd_addr_] && f Mark MST [bd_addr_])
                        f Mark MST [bd_addr_] = 0
                        ... ... ...
                        tcl.evalf("puts $namfile \"m -t %.15f -s %d -n m%d -c yellow -h
                        hexagon -X\"", clkoffset, bd_addr_, bd_addr_)
                else
                        if (f Mark MST [bd_addr_])
                                f Mark MST [bd_addr_] = 1
if ((ps!="NEW_CONNECTION_MASTER")) //
//estado anterior es = NEW_CONNECTION_MASTER?
        sprintf (lastest, "NEW_CONNECTION_MST")
        //lastest = NEW_CONNECTION_MST, disminuir tamaño de cadena
else
        sprintf (lastest, "%s", ps)
sprintf(anotacion, "%d en %.15f %s -> %s", bd_addr_, Scheduler::instance ().clock (), lastest,
nuevoest)
//escribir el formato a presentar como cometario en el NAM
... ... ...
tcl.evalf ("puts $namfile \"n -t %.15f -s %d -S DLABEL -l %s -L %s\"", clkoffset, bd_addr_,
nuevoest, lastest)
//instrucción escribir estado sobre los nodos
sec = sec + 1   //incrementar secuencia de próximo comentario
tcl.evalf ("puts $namfile \"v -t %.15f -e sim_annotation %.15f %d %s\"", clkoffset, clkoffset,
sec, anotacion)
//instrucción escribir comentario de forma secuencial  en la pantalla del NAM
```

Figura 3.4: *Algoritmo de Cambio de Estado*

Para lograr el posicionamiento inicial de los nodos, el cual se ejecuta cuando se invoca a la función ***setPos (X, Y)*** que se encuentra dentro de la librería ***wnode.cc,*** utilizaremos el siguiente formato:

tcl.evalf ("puts $namfile \"n -t * -s %d -x %.17f -y %.17f -Z 0 -z 1 -v circle -c black\"", getAddr(), x, y)

Donde ***getAddr (), x, y*** son el identificador del nodo, la posición en **X** y la posición en **Y** respectivamente.

Para realizar el efecto de movilidad durante la simulación se utilizará el siguiente formato:

tcl.evalf ("puts $namfile \"n -t %.9f -s %d -x %.17f -y %.17f -U %.6f -V %.6f -T %.6f\"", clkaux, bd_addr_, Coox, Cooy, Vx, Vy, tmp)

Donde ***Coox*** y ***Cooy*** son las posiciones iniciales desde donde el nodo empezará a moverse, ***Vx*** y ***Vy*** son las velocidades en **X** y la velocidad en **Y** respectivamente, ***tmp*** es el tiempo que dura el movimiento del nodo.

Ahora si el caso es poner una marca a los nodos que salgan del radio de cobertura de la piconet en la que se encuentran utilizaremos el siguiente formato:

tcl.evalf ("puts $namfile \"m -t %.15f -s %d -n md%d -c red -h hexagon\"", clkaux, bd_addr_, bd_addr_)

Donde la variable ***clkaux*** cumple la misma función que la variable ***clkoffset*** anteriormente descrita. Para remover estas marcas utilizaremos igual que en un caso anterior el siguiente formato:

tcl.evalf ("puts $namfile \"m -t %.15f -s %d -n md%d -c red -h hexagon -X\"", clkaux, bd_addr_, bd_addr_)

De igual manera la única diferencia entre los dos formatos anteriores es el parámetro ***–X*** **[*]** utilizado para remover las marcas de un nodo que poseía una.

Para implementar lo anterior en la función ***setdest ()***, se ha introducido una nueva variable global llamada ***mrknode [n] [m]***, teniendo **n** un rango [0....49] que representan los identificadores de los nodos y **m** un rango de [0....2], que almacena las siguientes variables: ***flagmrkON, flagmrkOFF, TIME***. Las cuales representan que el nodo debe ser marcado, que al nodo se le debe eliminar la marca y el tiempo en que ocurre la acción de marcar o quitarle la marca al nodo.

Podemos describir el segmento de código agregado a la función de movilidad mediante el siguiente algoritmo:

```
Setdest (destx, desty, destz, speed)
Tcl& tcl = Tcl :: instance ()
dst = 0, Coox = 0, Cooy = 0, Vx = 0, Vy = 0, tmp= 0, mst = 0, dstmst = 0, interval = 0,
Cooxaux = 0, Cooyaux = 0, dstaux = 0, nodeadd = 0
.....
if (f Coor [bd_addr_] = 1) then   //no es el primer movimiento?
//bd_addr_ es el identificador del nodo
        Coox = Pos X [bd_addr_]
        //asigno a Coox posición en X desde el arreglo Pos X
        Cooy = Pos Y [bd_addr_]
        //asigno a Cooy posición en Y desde el arreglo Pos Y
        Pos X [bd_addr_] = destx
        //almaceno la posición en X de destino como la actual del próximo salto
        Pos Y [bd_addr_] = desty
        //almaceno la posición en Y de destino como la actual del próximo salto
else                                        //es el primer movimiento
        Coox = node_->X_
        //asigno a Coox posición en X desde la estructura node_->X_
        Cooy = node_->Y_
        //asigno a Coox posición en Y desde la estructura node_->Y_
        Pos X [bd_addr_] = destx

        Pos Y [bd_addr_] = desty
        f Coor [bd_addr_] = 1               //se realizó el primer movimiento
.....
dst = sqrt ((destx – Coox) (destx – Coox) + (desty – Cooy) (desty – Cooy))
//calcular distancia hasta la posición de destino
Vx = speed * (destx – Coox)/dst
//calcular velocidad en X
Vy = speed * (desty – Cooy)/dst
//calcular velocidad en Y
tmp = dst/speed
//calcular duración del movimiento
... ... ...
```

```
... ... ...
... ... ...
tcl.evalf ("puts $namfile \"n -t %.9f -s %d -x %.17f -y %.17f -U %.6f -V %.6f -T %.6f\"",
clkaux, bd_addr_, Coox, Cooy, Vx, Vy, tmp)
//instrucción para realizar el movimiento del nodo
if (f es SLV [[bd_addr_]] = 1) then          // es esclavo?
        mst = f MST id [n]       //asignar identificador del nodo a variable entera
        dstmst = sqrt ((destx – Pos X [mst]) * (destx – Pos X [mst]) + (desty – Pos Y [mst]))
        //calcular la distancia final del nodo móvil a su respectivo maestro
        if (f Mark OUTR [bd_addr_] != 0) then  //está fuera de rango?
                if (dstmst > 10) then
                        f Mark OUTR [bd_addr_] = 1                    //fuera de rango
                        flagmrkON [bd_addr_] = 1
                        //activar la marca para ese nodo
                        do {
                         //obtener aproximaciones del tiempo que le toma
                        //salir del rango
                        interval = interval + 0.0001
                        //intervalos de 0.0001 segundos
                        Cooxaux = Coox + Vx  * interval
                        //posición en X obtenida con las aproximaciones de tiempo
                        Cooyaux = Cooy + Vy * interval
                        //posición en Y obtenida con las aproximaciones de tiempo
                        dstaux = sqrt ((Cooxaux - Pos X [mst]) *(Cooxaux - Pos X [mst]) +
                        (Cooyaux - Pos Y [mst]) *(Cooyaux - Pos Y [mst]))
                        //distancia al maestro obtenida con la posición estimada
                        } while (dstaux < 10)       //mientas este dentro de rango
                        TIME[bd_addr_] = Scheduler::instance ().clock () + interval
                        //tiempo aproximado que le toma salir de rango
        else
                if (dstmst <= 10) then
                        f Mark OUTR [bd_addr_] = 0  //dentro del rango
                        flagmrkOFF [bd_addr_] = 1
                         //desactivar la marca para ese nodo
                        do {
                         //obtener aproximaciones del tiempo que le toma
                        //salir del rango
                        interval = interval + 0.0001
                        Cooxaux = Coox + Vx  * interval
                        Cooyaux = Cooy + Vy * interval
                        dstaux = sqrt ((Cooxaux - Pos X [mst]) *(Cooxaux - Pos X [mst]) +
                        (Cooyaux - Pos Y [mst]) *(Cooyaux - Pos Y [mst]))
                        } while (dstaux > 10)
                        TIME[bd_addr_]  = Scheduler::instance ().clock () + interval
else            //es maestro
        for (nodeadd = ; nodeadd < 50; nodeadd ++)
                if ((f MSTid[6] == bd_addr_) && f CONNET[nodeadd] && f es
                MST[bd_addr_]) then
                //identificar los nodos que están conectados al maestro
                dstmst = sqrt ((destx – Pos X [mst]) * (destx – Pos X [mst]) + (desty – Pos Y
                [mst]))
                //calcular las distancias de los nodos al maestro móvil
                if (f Mark OUTR [nodeadd] != 1) {
                if (dstmst > 10) then
                        f Mark OUTR [nodeadd]= 1          //fuera de rango
                        flagmrkON [nodeadd] = 1
                        do {
                        interval = interval + 0.0001
                        Cooxaux = Coox + Vx  * interval
                        Cooyaux = Cooy + Vy * interval
```

```
                dstaux = sqrt ((Cooxaux - Pos X [nodeadd]) *(Cooxaux - Pos X
                [nodeadd]) + (Cooyaux - Pos Y [nodeadd]) *(Cooyaux - Pos Y
                [nodeadd]))
                } while (dstaux < 10)
                TIME[nodeadd] = Scheduler::instance ().clock () + interval
        else
                if (dstmst <= 10) then
                        f Mark OUTR [bd_addr_]= 0       //fuera de rango
                        flagmrkON [bd_addr_] = 1
                        do {
                        interval = interval + 0.0001
                        Cooxaux = Coox + Vx  * interval
                        Cooyaux = Cooy + Vy * interval
                        dstaux = sqrt ((Cooxaux - Pos X [nodeadd]) *(Cooxaux - Pos
                        X [nodeadd]) + (Cooyaux - Pos Y [nodeadd]) *(Cooyaux -
                        Pos Y [nodeadd]))
                        } while (dstaux > 10)
                        TIME[nodeadd] = Scheduler::instance ().clock () + interval
```

Figura 3.5: *Función SetDest()*

Además en esta función almacenamos información de cuales son los nodos a los cuales se les debe poner la marca de fuera de rango o a cuales hay que quitárselas cuando entren en el rango permitido y en que momento esto ocurre, esto se lo realiza mediante aproximaciones del tiempo en que ocurren la salida y el ingreso del nodo en el rango permitido, esta información se almacena en el arreglo ***mrknode [n] [m].*** La ejecución de estas acciones se realizan en la función ***dump (),*** la cual se encuentra en la librería ***baseband.cc,*** como se detalla en el siguiente algoritmo:

```
dump ()
.....
.....
 for (nodeident = 0; nodeident < 50; nodeident++)
        if (flagmrkON [nodeident] == 1) then
        //verificar a que nodos se les debe colocar la marca
        //de fuera de rango
                if (Scheduler::instance().clock() > TIME[nodeident])
                //verificar el tiempo al cual se colocara la marca
                        tcl.evalf ("puts $namfile \"m -t %.9f -s %d -n md%d -c red -h
                        hexagon\"", Scheduler::instance().clock(), nodeident, nodeident);
                        //instrucción para colocar la marca
                        flagmrkON [nodeident]  = 0;
                         // Poner Mark SLV
... ... ...
... ... ...
... ... ...
```

```
for (nodeident = 0; nodeident < 50; nodeident++)
        if (flagmrkOFF [nodeident] == 1)
        //verificar a los nodos a los cuales se les debe quitar la marca
        //de fuera de rango
                if (Scheduler::instance().clock() > TIME[nodeident])
                //verificar el tiempo al cual se quitara la marca
                tcl.evalf ("puts $namfile \"m -t %.9f -s %d -n md%d -c red -h hexagon -X\"",
                Scheduler::instance().clock(), nodeident, nodeident);
                //instrucción para quitar la marca
                flagmrkOFF [nodeident] = 0;
                 // Sacar Mark SLV
```

Figura 3.6: *Función Dump()*

Se realizó este proceso en esta función debido a que esta se actualiza constantemente, esto nos permite realizar la acción justo en el momento en que esta debe ocurrir por el contrario la función ***setdest ()*** no nos permitía realizar esto debido a que solo se ejecuta en el momento en que esta es invocada impidiéndonos realizar la acción en tiempo real.

3.3 FUNCIONAMIENTO GENERAL

Nuestro proyecto es una adaptación para el uso del NAM en la visualización de las simulaciones de los casos típicos de redes Bluetooth. Las mismas que serán implementadas a manera de prácticas para el laboratorio de telecomunicaciones.

Las modificaciones que se realizaron permitieron que la librería adquiera un ambiente compatible con NAM, puesto que antes su ambiente era muy pobre y no daba detalles que son buenos resaltar para ir conociendo los procesos y detalles importantes dentro de las redes Bluetooth.

La librería cuenta con varios comandos, dependiendo de los casos se vayan a simular es necesario conocer el escenario y plantearlo adecuadamente.

Así mismo debe tenerse en cuenta que a medida que el escenario involucra más nodos, la visualización de la misma tendera a ser más compleja de entender. Es por esta razón que implementamos el uso de ciertas variables en la visualización de la simulación para que actúen como "notificadores de eventos" y de "estados", teniendo así una orientación sin perder datos de importancia. También de implementó la detección de rangos de cobertura en la interfaz gráfica.

3.3.1 DIAGRAMAS DE BLOQUE DEL PROYECTO

A continuación se muestra en la Figura 3.7 el diagrama de un nodo inalámbrico regular dentro del NS2, como podemos observar un nodo consta de varias variables que tienen funciones específicas para que este trabaje de la forma que se requiere.

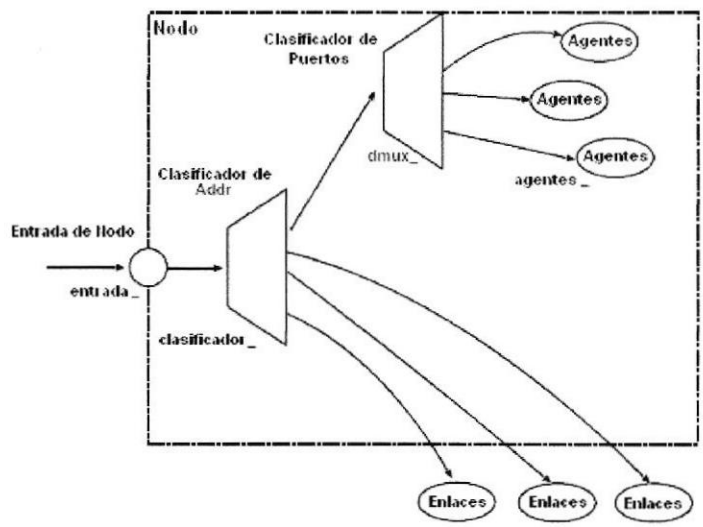


Figura 3.7: *diagrama de un nodo en NS2*

Como podemos apreciar los nodos constan de un identificador, y varios puertos donde se realizan los enlaces y se combinan con el agente específico, sea de ruteo o de transporte.

En el caso de UCBT los nodos tienen una forma especial puesto que se pretende usen la pila de protocolos básica de todo dispositivo Bluetooth, es así como observamos en la Figura 3.8 el diagrama de los nodos en UCBT.

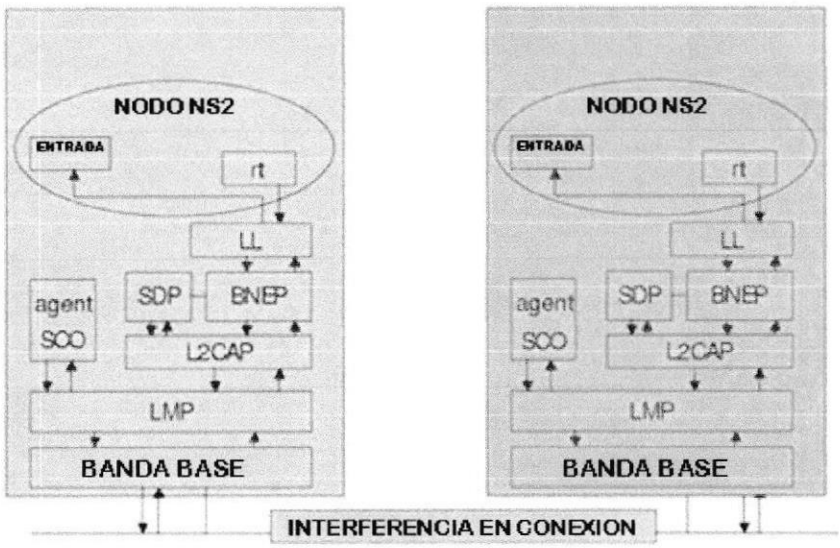


Figura 3.8: *Diagrama de los nodos Bluetooth en UCBT*

Como podemos apreciar en UCBT el nodo cuenta con las capas banda base, LMP, L2CAP, SDP, BNEP y SCO de la especificación PAN de Bluetooth y las capas LL, RT y de entrada correspondientes a las de un nodo inalámbrico en NS2, es de esta manera que los agentes y enlaces que manejan están dados por el tipo de paquetes de cada una de las capas Bluetooth mencionadas, proporcionando así el esquema del funcionamiento de cada nodo Bluetooth.

CAPÍTULO 4:

4. IMPLEMENTACIÓN DEL PROYECTO

4.1 SIMULACIÓN DE PROCEDIMIENTOS DE CONEXIÓN

Los procedimientos de conexión que se simularán son los siguientes:

- Inquiry o Descubrimiento de dispositivos dentro del rango de cobertura.
- Paging o Establecimiento de una red Bluetooth entre dispositivos dentro del rango de cobertura.

4.1.1 SIMULACIÓN DEL PROCEDIMIENTO DE INQUIRY

Para iniciar cualquier tipo de comunicación en redes Bluetooth, primero deben de conocerse los dispositivos que estén dentro de un área de cobertura específica, en el caso de las simulaciones es de aproximadamente un radio de 10 metros. A este procedimiento de descubrimiento de dispositivos cercanos, se lo conoce con el nombre de Inquiry.

Como se mencionó en la sección 2.1.5, en este procedimiento, un nodo buscará dispositivos cercanos que estén en modo detectable, por lo tanto están recibiendo peticiones de Inquiry utilizando el estado de

Inquiry Scan. Aquellos dispositivos que no estén en Inquiry Scan, simplemente no serán detectados por el dispositivo que los busca. Los nodos que estén en modo detectable enviarán paquetes de Inquiry Response al dispositivo que los busca, como se muestra en la figura 4.1.

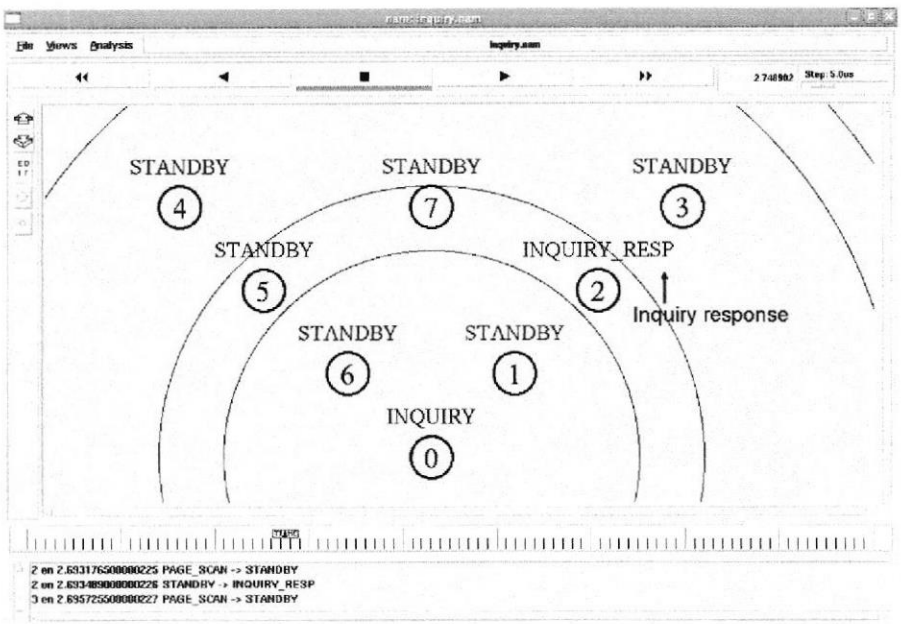


Figura 4.1: *Respuesta de un dispositivo que está en modo detectable.*

Los resultados obtenidos en esta simulación son conformes a la práctica 1 "Simulación de procedimientos básicos para la conexión de dispositivos Bluetooth basados en NS2/UCBT/NAM", procedimiento 5.1 "Simular un procedimiento de Inquiry y observar los cambios de estados de los nodos", del apéndice B.

4.1.2 SIMULACIÓN DEL PROCEDIMIENTO DE PAGING

Para ejecutar este procedimiento de conexión entre dispositivos Bluetooth, también conocido como Paging, se necesita que los dispositivos a conectarse se conozcan, razón por la cual debió existir un procedimiento de Inquiry previo entre ellos.

En este procedimiento, tendremos un dispositivo demandante que estará en estado de Page buscando dispositivos que se encuentren en estado conectable o Page Scan. Estos dispositivos enviarán respuestas o Page Responses al dispositivo demandante, informándole que están listos para continuar con el proceso de conexión. En la figura 4.2, observamos al nodo demandante en estado de Page y a los nodos detectados en estado de Page Scan o Conectable.

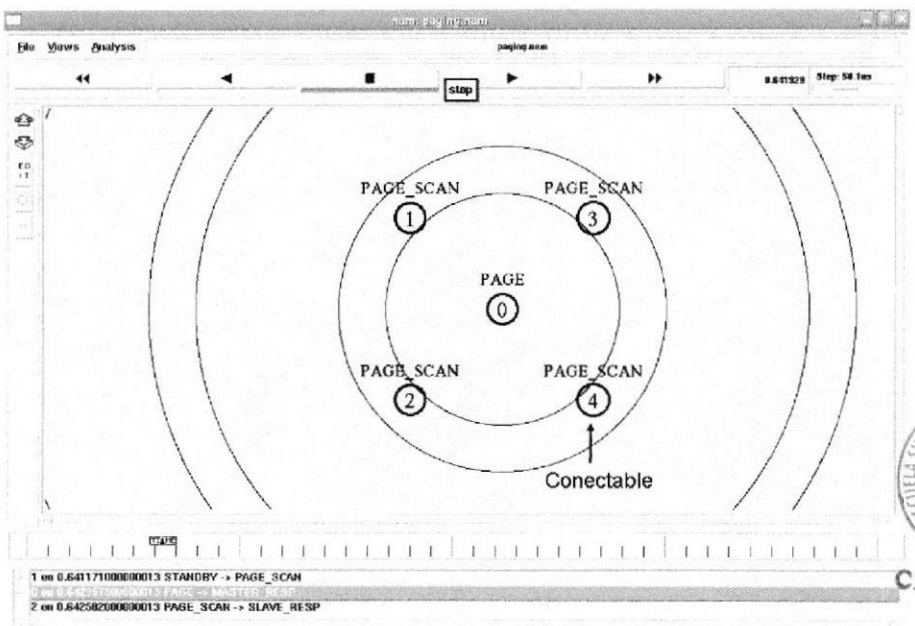


Figura 4.2: *Dispositivos en estado de Page y Page Scan respectivamente.*

Una vez que el dispositivo demandante reciba estas respuestas, se procederá a formar un reloj global y un esquema de saltos de frecuencia único, con lo cual se iniciará la piconet, dando como resultado un dispositivo maestro (el dispositivo demandante) y varios

dispositivos esclavos (los dispositivos que aceptan la conexión del demandante).

Los resultados obtenidos en esta simulación son conformes a la práctica 1 "Simulación de procedimientos básicos para la conexión de dispositivos Bluetooth basados en NS2/UCBT/NAM", procedimiento 5.2 "Simular un procedimiento de Paging y observar los roles que adquiere los nodos", del apéndice B.

4.2 SIMULACIÓN DE PICONETS

Una piconet constituye la forma más básica de conexión en una red Bluetooth. En pocas palabras, una piconet es una red de varios dispositivos conectados vía Bluetooth sincronizados por un reloj común, obtenido del dispositivo maestro.

Este tipo de conexión fue descrito anteriormente en la sección 2.2.2. Las piconets pueden ser consideradas como conexiones punto a punto cuando se trata de dos dispositivos únicos y como conexiones punto a multipunto cuando existen más de dos dispositivos conectados a la vez.

Una piconet puede empezar con un nodo maestro y un único nodo esclavo, pero a medida que aumenten las necesidades o según las aplicaciones de la red, existe la posibilidad que más nodos pasen a formar parte de una misma piconet. El número máximo de nodos esclavos activos en una piconet es de siete sin contar al nodo maestro, pero pueden existir hasta

doscientos cincuenta y cinco nodos esclavos en modo estacionado por si la red en algún momento los requiere. A continuación mostramos una piconet con siete nodos esclavos activos y un nodo no conectado.

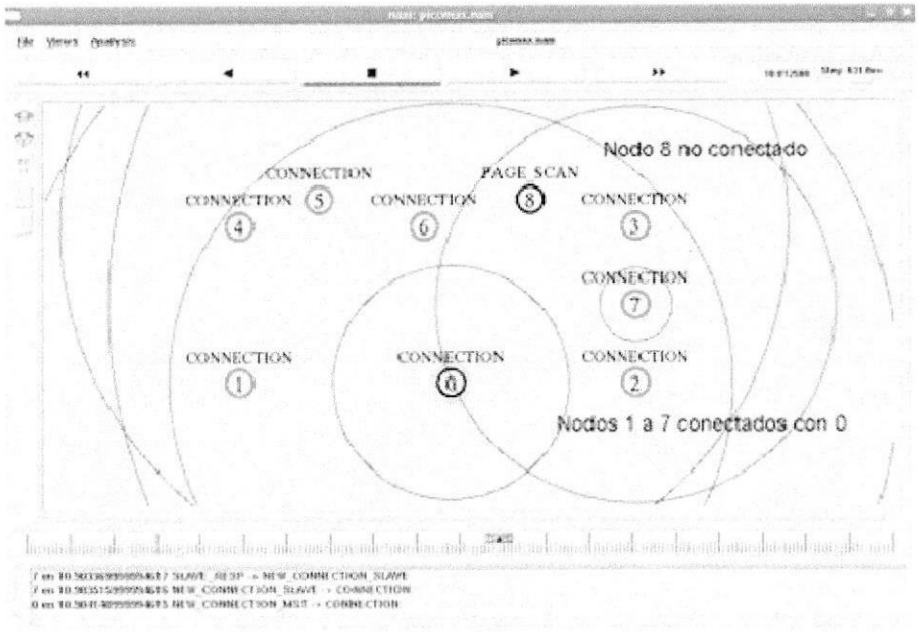


Figura 4.3 *Piconet básica.*

Los resultados obtenidos en esta simulación son conformes a la práctica 1 "Simulación de procedimientos básicos para la conexión de dispositivos Bluetooth basados en NS2/UCBT/NAM", procedimiento 5.3 "Simular una piconet con la cantidad máxima de nodos conectados y con una transferencia de archivos bajo una aplicación TCP", del apéndice B.

4.3 SIMULACIÓN DE TRANSFERENCIA DE PAQUETES ENTRE NODOS.

Para que exista un tipo de aplicación en lo que respecta a Bluetooth, primero se necesita que los nodos entre los cuales se requiera establecer la aplicación estén conectados de alguna manera, ya sea mediante conexión directa entre ellos (maestro - esclavo) o indirecta (esclavo - maestro - esclavo) dependiendo de la red que se haya formado.

Con cada aplicación se deberá establecer una comunicación en todas las capas de la pila de protocolos de Bluetooth que sean necesarias para soportar la misma. Las aplicaciones simuladas entre nodos Bluetooth son FTP y CBR.

Los protocolos TCP y UDP, ampliamente utilizados en aplicaciones que tienen que ver con transferencias a nivel de capa de transporte en redes IP, son nuestro enfoque para las simulaciones. Para establecer una comunicación de este tipo, los nodos previamente establecieron la comunicación entre sus capas inferiores, para lograr esto se intercambiaron paquetes de las capas Banda Base, LMP, L2CAP, BNEP, IP.

Un ejemplo típico en TCP es una aplicación FTP que se puede dar entre dispositivos Bluetooth. En esta aplicación se usa un esquema de comunicación donde se asegura que los paquetes hayan llegado a su destino sin errores. De existir alguno, se reenvía el paquete que no llegó al destino.

Esto lo realiza mediante paquetes de ACK (donde se especifica si algún paquete no fue recibido). Con este esquema se asegura que los datos lleguen correctamente a su destino. En la figura 4.4, vemos que la transferencia se ejecuta desde el nodo 0 al nodo 1, como se puede apreciar ambos nodos intercambian paquetes de información y reconocimiento ACK.

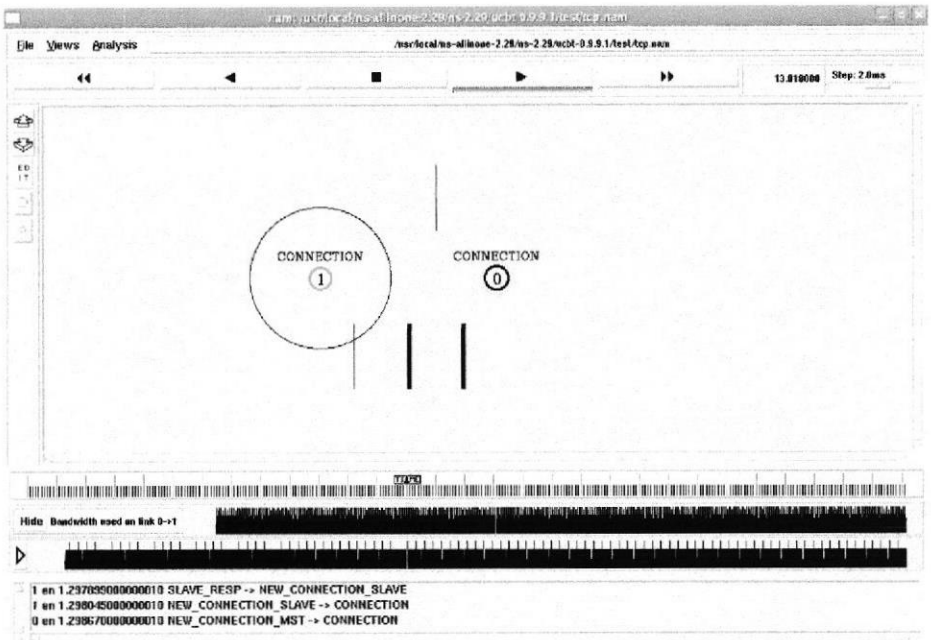


Figura 4.4 *Transferencia TCP*

El protocolo UDP suele ser utilizado para las transferencias de datos a velocidad constante (CBR) que tiene variedad de aplicaciones sobre todo para comunicaciones de voz, donde los paquetes tienen un tamaño ya predefinido según la compresión que se utilice. Es por esta razón que este tipo de paquetes no son recuperables, si alguno no llegó a su destino, no será reenviado por el origen.

En la figura 4.5 se muestra una aplicación CBR entre los nodos 0 y 1. Nótese que la transferencia de datos se realiza desde el nodo 0 hacia el nodo 1 como se puede apreciar en la parte inferior de la figura. Además vemos que no existe respuesta del nodo 1 hacia el nodo 0.

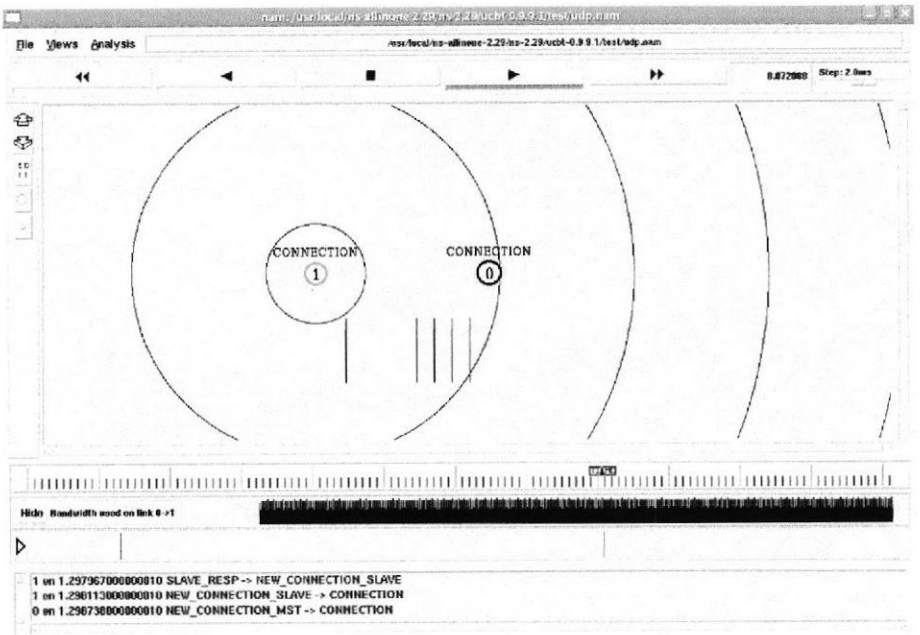


Figura 4.5 *Transferencia UDP*

Los resultados obtenidos en esta simulación son conformes a la práctica 2 "Transferencia de paquetes en diferentes topologías de redes Bluetooth basados en NS2/UCBT/NAM", procedimientos 5.2 y 5.3, del apéndice B.

4.4 SIMULACIÓN DE SCATTERNETS

La forma de conexión más compleja en una red Bluetooth lo representan las scatternets, las cuales son redes que interconectan dos o más piconets entre sí mediante el uso de nodos que participan en varias piconets (nodos PMP). Este tipo de nodos servirán como puente entre las piconets que se interconecten.

En las simulaciones que presentamos, se puso énfasis en los elementos necesarios para poder establecer una scatternet como lo son los nodos PMP y el procedimiento de intercambio de roles de un nodo. Cabe recalcar que este tipo de nodos pueden tener varios roles según la piconet en la cual

estén participando. Puede ser nodo maestro de una piconet y a la vez nodo esclavo en la otra.

A continuación se muestra una scatternet básica de tres piconets diferentes con sus respectivos nodos maestros.

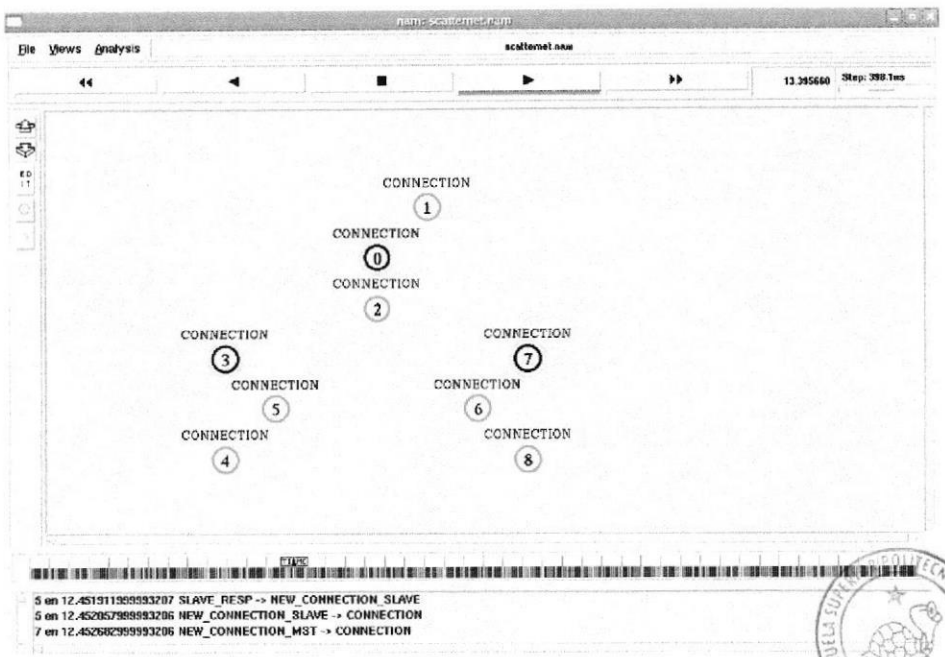


Figura 4.6 *Scatternet básica.*

Los resultados obtenidos en esta simulación son conformes a la práctica 2 "Transferencia de paquetes en diferentes topologías de redes Bluetooth basados en NS2/UCBT/NAM", procedimiento 5.4 "Simular una red Bluetooth avanzada (Scatternet)", del apéndice B.

4.5 COSTOS DE IMPLEMENTACIÓN

Los costos específicos de la implementación del proyecto, tomando en cuenta los factores que ya se han venido mencionando como es el simulador de licencia libre, el sistema operativo de código abierto, entre otros, se reflejan en la tabla 4.1.

Ítem	Costo licencia	Costo
Linux con todos los paquetes de herramientas de desarrollo, editores y la interfaz gráfica GNOME	$ 0,00	$ 0,00
Simulador NS2	$ 0,00	$ 0,00
Librería UCBT	$ 0,00	$ 0,00
Computador P4 2.4 GHz 1GB RAM HDD de 80GB (recomendado)	No aplica	*$ 800,00*
Costo de desarrollo del proyecto (José Landa, José Jimbo, Luis Salazar)	No aplica	$ 1.200,00
TOTAL	**$ 0.00**	**$ 2000,00**

Tabla 4.1 *Costos de implementación del proyecto.*

Cabe recalcar que se desarrollaron e implementaron las modificaciones necesarias a la librería UCBT para que así tenga un ambiente más amigable y las simulaciones sean más didácticas, es por esta razón que dentro de los costos estamos considerando todo ese tiempo, esfuerzo y dedicación que se emplearon en horas de Internet, en la investigación de los lenguajes que utiliza la librería y el sistema operativo, los conocimientos del estándar, cursos de programación y asuntos relacionados inclusive al sistema operativo Linux. Este valor es una simple representación del trabajo realizado. Nótese, que básicamente el costo de licencia es de $0.00, lo que representa un ahorro significativo para las instituciones que lo implementen.

CONCLUSIONES Y RECOMENDACIONES

CONCLUSIONES

1. Las modificaciones realizadas a ciertas funciones de la librería UCBT nos han permitido corroborar el correcto funcionamiento de la especificación Bluetooth como por ejemplo los estados por los que tiene que someterse un dispositivo para establecer una comunicación, la movilidad de dos dispositivos conectados para demostrar que el rango de cobertura no debe sobrepasar los diez metros y la transferencia de información entre un maestro y un esclavo.

2. Las prácticas didácticas propuestas para su uso en el laboratorio de telecomunicaciones, se han desarrollado con la finalidad de dar al estudiante una herramienta de fácil comprensión y entendimiento con lo que respecta al alcance de este proyecto de graduación.

3. El hecho de adaptar a la librería una interfaz gráfica permitió que, la revisión de los resultados obtenidos y ciertos acontecimientos que ocurren durante el transcurso de las simulaciones, sean aprovechados por el investigador para un mejor entendimiento de los procesos y las limitaciones que posee la tecnología Bluetooth.

4. Debido a la falta de información sobre esta librería en publicaciones científicas, este trabajo servirá de ayuda y guía para aquellas personas que trabajan o realizan alguna investigación relacionada con la tecnología Bluetooth utilizando el Network Simulator 2 (NS2) en conjunto con la librería UCBT.

RECOMENDACIONES

1. Aprovechando la ventaja de que la herramienta usada para realizar las simulaciones en esta investigación es de código abierto, se recomienda que se utilice el contenido de este trabajo para contribuir al desarrollo de la librería con el fin de obtener mejores resultados. Además, todo aporte realizado a esta librería debería estar correctamente documentado para facilitar el entendimiento de los cambios implementados. Estas modificaciones podrían ser publicadas utilizando la lista de correos (mailing list), para que la comunidad que utiliza esta herramienta tenga conocimiento de la existencia de una versión mejorada.

2. Se recomienda el uso de esta herramienta a las personas que quieran profundizar en el funcionamiento de la tecnología Bluetooth, ya que nos muestra de forma detallada los procesos que se realizan para poder establecer una conexión o una transferencia de información entre dispositivos, así como un registro detallado de eventos, el cual nos brinda información sobre lo que esté ocurriendo durante el tiempo que dura cada simulación.

3. Los estudiantes que llegaran a realizar las prácticas de laboratorio incluidas en este documento, deberían recibir una capacitación previa al funcionamiento del simulador, la librería y el sistema operativo. La información básica de estos temas está incluida en el apéndice.

4. Con la información que se provee en los apéndices de este documento, los tutores deberían de estar en capacidad de realizar diferentes escenarios a los que se encuentran planteados en las prácticas y así incrementar el número de versiones de cada una de ellas.

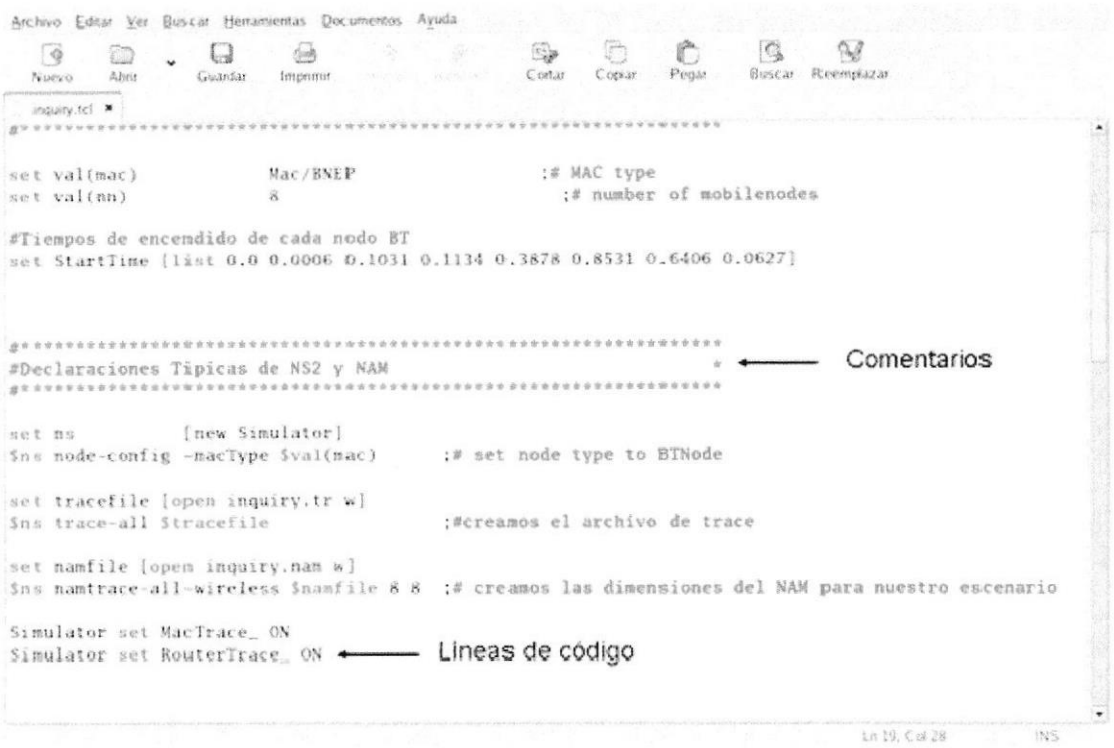
APENDICES

APÉNDICE A.

Guías para comenzar a trabajar con NS2, UCBT y NAM.

NS2 es un simulador de redes de código abierto en el que se pueden realizar diferentes escenarios de redes tanto alámbricas como inalámbricas. Este simulador utiliza comandos que son escritos en un archivo llamado script, donde se especifican todas las variables, nodos, dispositivos, conexiones, tipos de tráficos, entre otros.

La siguiente figura es un ejemplo de este archivo:



Script para simulación en NS2

La creación de un script es sencilla y para ello solo necesitamos escribir las sentencias en lenguaje tcl y siguiendo la estructura adecuada en cualquier editor de texto. Una de las ventajas de usar el lenguaje tcl es que no se necesita definir el tipo de variable que se utilizaran, es por ello que si revisamos un archivo tcl no encontraremos tipos de datos entero, reales, etc. Lo único que debemos definir son los agentes que se utilizaran en la simulación. La declaración de los agentes y la estructura que se debe seguir al momento de la creación de un script se detallan a continuación:

set** variable_simulador **[new Simulator]
esta sentencia realiza la declaración de la variable que identificará la instancia del simulador o también conocida como variable del simulador. Hay que tener en cuenta que si se utilizan constantes dentro del script, éstas deben ser declaradas con sus respectivos valores justo antes de especificar esta sentencia. Lo podemos notar en la figura anterior.

set** variable_trace **[open** nombre_archivo.**tr w]

$variable_simulador ***trace-all*** *$variable_trace*
estas líneas crean el archivo de trazas disponible para escritura, además le indican al simulador que todos las trazas generadas se deben almacenar en el archivo que se creó anteriormente.

A continuación se procede a la creación de los nodos con su respectivas características, dependiendo del tipo de librería que se esta utilizando, seguido por la definición de los enlaces, tipo de tráfico y declaración de colas en caso de necesitarse en la simulación. Luego procedemos a definir la organización de los eventos. La explicación de esta sección se detallará más adelante en esta guía.

Para finalizar todo script debe realizarse un procedimiento en el cual se detallarán ciertas acciones que se deben ejecutar antes de salir del simulador. Este procedimiento se detalla a continuación:

```
proc finish {} {
        global node variable_simulador  variable_trace variable_nam
        $ variable_simulador flush-trace
        $node(0) print-all-stat
        close $ variable_nam
        close $ variable_trace
        exec nam nombre_archivo.nam &
        exit 0
}
$ns run
```

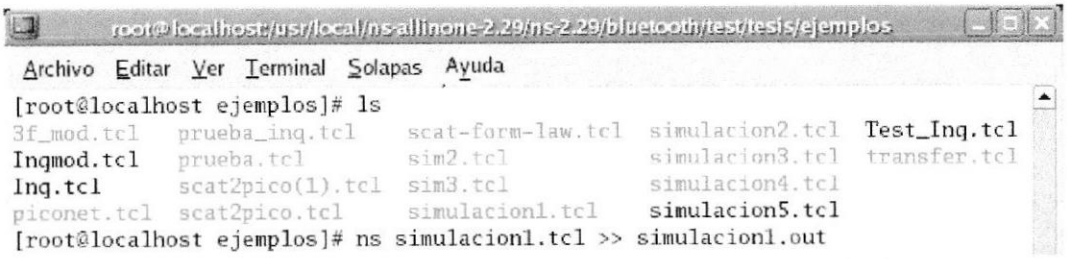
Procedimiento de finalización

En este procedimiento se ejecutan varias acciones, entre las principales están el cierre de los archivos *.nam* y *.tr* que se abrieron al inicio de la simulación, para ello al inicio del procedimiento se definen las variables que representan a estos archivos como globales.

También podemos realizar ciertas acciones especiales para ciertos nodos así como también escribir sentencias que se ejecutarán en la línea de comando del entorno donde se ejecute la simulación. Para este caso es una ventana de terminal muy parecida a la ventana de DOS en los sistemas operativos Windows, esto se logra utilizando el comando *exec*. Por último especificamos que al terminar de procesar el contenido del archivo tcl, la ventana de terminal permanezca abierta por el comando *exit 0*.

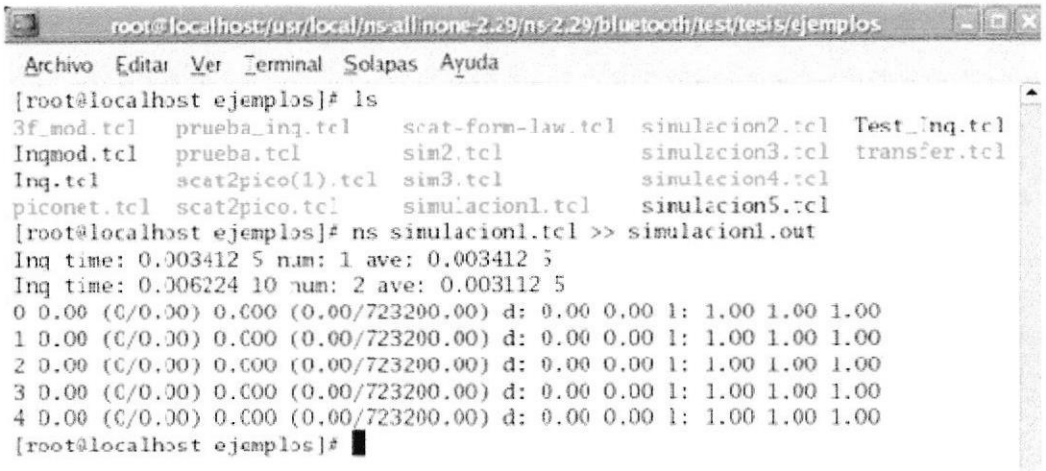
Una vez que tenemos listo nuestro script procedemos a la ejecución de este archivo. Para ejecutar una simulación, primero debemos abrir una ventana de terminal donde escribimos la siguiente sentencia:

ns *nombrearchivo.tcl >> nombrearchivo.out*



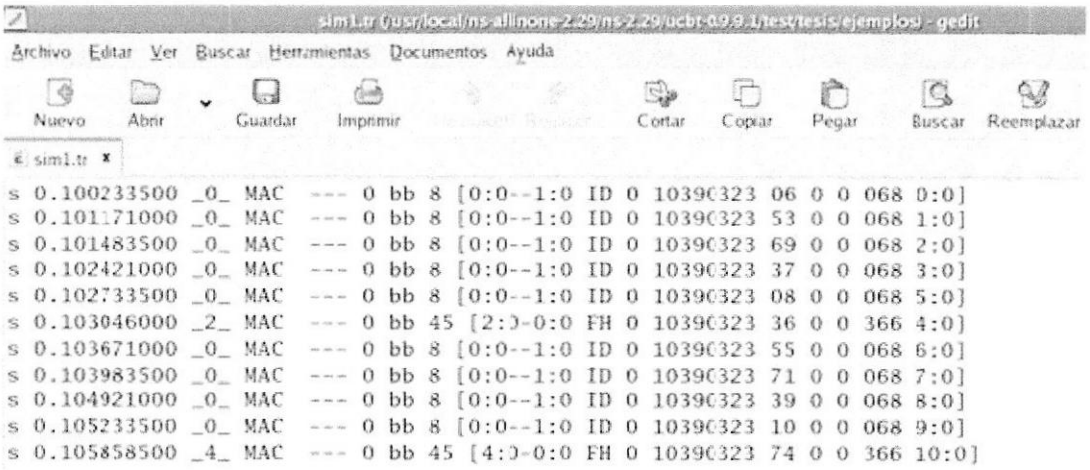
Pantalla de terminal con sentencias para ejecutar simulación.

Para lograr esto el archivo que contiene la simulación que se va a ejecutar, en este caso identificado como *nombrearchivo*, debe estar contenido en la misma carpeta que indica la ruta en la pantalla del terminal.



Pantalla de terminal en un instante después de terminar la simulación

Una vez que el simulador termina de procesar el script, pasamos a la revisión de los resultados, estos se guardan en archivos con extensión *.tr* denominados archivos de traza y *.out* denominados archivos de salida. La siguiente figura es un ejemplo del archivo de trazas, el archivo *sim1.tr*:



Archivo sim1.tr visualizado en el editor de texto.

Estos archivos de trazas siguen un formato específico, para poder ser analizados se necesita conocer el significado de cada uno de los parámetros que se encuentran en las

líneas de este archivo. El formato de estas trazas comienza con uno de cuatro caracteres. Este es seguido por una bandera ó pares de valores los cuales son detallados a continuación.

Evento	Abreviatura	Tipo	Valor
Eventos Normales	r: Recibe d: Caído e: Error +: En cola -: Sale de cola	%g %d %d %s %d %s %d %d.%d %d.%d %d %d	
		doble	tiempo
		entero	Nodo fuente
		entero	Nodo destino
		caracteres	Nombre del paquete
		entero	Tamaño del paquete
		caracteres	Banderas
		entero	ID para el flujo
		entero	Dirección de la fuente
		entero	Dirección del destino
		entero	Numero de secuencia
		entero	ID único del paquete

Formatos de trazas normales.

La tabla que lista la información adicional de trazas inalámbricas no tiene la columna abreviatura, esta información es añadida al final de las trazas inalámbricas regulares.

Evento	Tipo	Valor
Trazos TCP	%d 0x%x %d %d	
	entero	Número de Ack
	hexadecimal	Banderas
	entero	Longitud de cabeceras
	entero	Longitud de la dirección del puerto

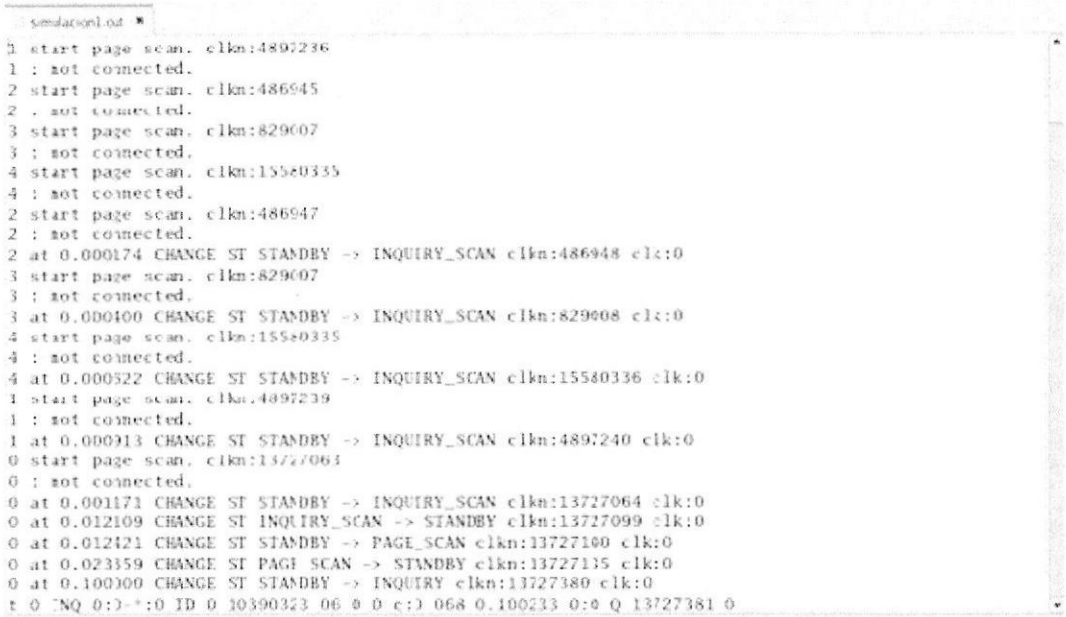
Formato de trazas adicionales.

Evento	Abreviatura	Bandera	Tipo	Valor
Eventos inalámbricos	s: Enviado r: Recibido d: Caído f: Reexpedir	-t	doble	tiempo (* para ajustes globales)
		-Ni	entero	ID del nodo
		-Nx	doble	Coordenada X del nodo
		-Ny	doble	Coordenada Y del nodo
		-Nz	doble	Coordenada Z del nodo
		-Ne	doble	Nivel de energía del nodo
		-Nl	caracteres	Nivel de trazo de la red (AGT, RTR, MAC, bb, etc.)
		-Nw	caracteres	Razón de la caída
		-Hs	entero	ID del nodo fuente del salto
		-Hd	entero	ID del nodo destino del salto , -1, -2

		-Ma	hexadecimal	Duración
		-Ms	hexadecimal	Dirección Ethernet de la fuente
		-Md	hexadecimal	Dirección Ethernet del destino
		-Mt	hexadecimal	Tipo de Ethernet
		-P	caracteres	Tipo de paquete (arp, dsr, imep, tora, etc.)
		-Pn	caracteres	Tipo de paquete (cbr, tcp)

Formato de trazas inalámbricas.

Otra de las fuentes donde podemos obtener información es el archivo de salida de la simulación. La siguiente figura muestra un ejemplo de este archivo de salida, el archivo *simulacion1.out*:



Archivo simulacion1.out visualizado en el editor de texto.

Este archivo tiene un formato especifico según la librería que se esté utilizando para simular los escenarios, en este caso la simulación corresponde a la de una red Bluetooth, por lo cual este archivo tiene un formato acorde con la librería UCBT.

Las trazas más importantes son las que hacen referencias a los eventos de transmisión y recepción. Estos son identificados por medio de la letra con la que empieza cada traza para el caso de transmisión "t" y para recepción "r". Los formatos de estos eventos se detallarán en la siguiente tabla:

Evento	Tipo	Bandera	Valor
Trazas archivo de salida	%c %d %s %d:%d-%d:%d		
	carácter	stat	Estado de transmisión
	entero	ad	Identificador del nodo
	caracteres	st	Estado del nodo
	entero	sender	Identificador del emisor
	entero	srcTxSlot	Slot en emisión

entero	receiver	Identificador del receptor
entero	dstTxSlot	Slot en recepción
%s %d %d %.02d %d %d c:%d %.03d %f %d:%d %s %d %d		
caracteres	packet_type_str_short(type)	Tipo del paquete
entero	lt_addr_	Identificador del nodo
entero	ac	
entero	fs_	Secuencia de salto de frecuencia
entero	arqn	
entero	seqn	
entero	transmitCount	
entero	size	Tamaño de paquete
real	ts()	Tiempo
entero	pid	Tamaño de la carga útil
entero	seqno	Secuencia
caracteres	comment()	Comentario
doble	clk	
entero	extinfo	

Formato del archivo de salida.

El NAM es la herramienta que utilizaremos para la visualización de las simulaciones que se ejecuten con el NS2, siempre y cuando la librería que se utilice tenga compatibilidad para crear el archivo que es interpretado por el NAM, archivos *.nam*, los cuales poseen todos los detalles de la simulación que han sido compiladas por el simulador.

A continuación tenemos un ejemplo de un archivo *.nam*.

```
sim1.nam *
V -t * -v 1.0a5 -a 0
W -t * -x 670 -y 670
A -t * -n 1 -p 0 -u 0xffffffff -c 31 -a 1
A -t * -h 1 -m 2147483647 -s 0
v -t 0.000172999993083 -e sim_annotation 0.000172999993083 0 2 en 0.000174000000000 STANDBY ->
INQUIRY_SCAN
n -t 0.000172999993083 -s 2 -S DLABEL -l INQUIRY_SCAN -L STANDBY
v -t 0.000399000011384 -e sim_annotation 0.000399000011384 1 3 en 0.000400000000000 STANDBY ->
INQUIRY_SCAN
n -t 0.000399000011384 -s 3 -S DLABEL -l INQUIRY_SCAN -L STANDBY
v -t 0.000621000015780 -e sim_annotation 0.000621000015780 2 4 en 0.000622000000000 STANDBY ->
INQUIRY_SCAN
n -t 0.000621000015780 -s 4 -S DLABEL -l INQUIRY_SCAN -L STANDBY
v -t 0.000912000017706 -e sim_annotation 0.000912000017706 3 1 en 0.000913000000000 STANDBY ->
INQUIRY_SCAN
n -t 0.000912000017706 -s 1 -S DLABEL -l INQUIRY_SCAN -L STANDBY
v -t 0.001169999944977 -e sim_annotation 0.001169999944977 4 0 en 0.001171000000000 STANDBY ->
INQUIRY_SCAN
n -t 0.001169999944977 -s 0 -S DLABEL -l INQUIRY_SCAN -L STANDBY
v -t 0.012108599767089 -e sim_annotation 0.012108599767089 5 0 en 0.012108500000000 INQUIRY_SCAN -
> STANDBY
n -t 0.012108599767089 -s 0 -S DLABEL -l STANDBY -L INQUIRY_SCAN
v -t 0.012421100400388 -e sim_annotation 0.012421100400388 6 0 en 0.012421000000000 STANDBY ->
PAGE_SCAN
n -t 0.012421100400388 -s 0 -S DLABEL -l PAGE_SCAN -L STANDBY
v -t 0.023358600214124 -e sim_annotation 0.023358600214124 7 0 en 0.023358500000000 PAGE_SCAN ->
STANDBY
n -t 0.023358600214124 -s 0 -S DLABEL -l STANDBY -L PAGE_SCAN
```

Archivo sim1.nam visualizado en el editor de texto.

El archivo *.nam* está conformado por un conjunto de líneas las cuales poseen parámetros, el formato general para un traza del NAM es una letra seguida por uno o más pares de valores. En la siguiente tabla se detallan estos parámetros, hay que tener en cuenta que no todos estos serán usados siempre:

Evento	Abrev.	Bandera	Tipo	Valor
Nodo	n	-t	tiempo	Tiempo
		-s	entero	Nodo ID
		-u	doble	Velocidad X
		-U	doble	Velocidad X
		-V	doble	Velocidad Y
		-v	Forma	Forma (circle, box, hexagon)
		-c	color	Color
		-z	doble	Tamaño de nodo
		-a	entero	Dirección
		-x	doble	Posición X
		-y	doble	Posición Y
		-Z	doble	Posición Z (0)
		-i	color	Color de la etiqueta
		-b	caracteres	Etiqueta
		-l	caracteres	Etiqueta
		-o	color	Color previo
		-S	caracteres	Estado (UP, DOWN, COLOR)
		-L	caracteres	Etiqueta previa
		-p	caracteres	Posición de la etiqueta
		-P	caracteres	Posición previa de la etiqueta
		-i	color	Color del interior de la etiqueta
		-I	color	Color previo del interior de la etiqueta
		-e	color	Color de la etiqueta
		-E	color	Color previo de la etiqueta
		-T	doble	Duración del movimiento
		-w	bandera	Nodo inalámbrico
Enlace	l	-t	tiempo	Tiempo
		-s	entero	ID de la fuente
		-d	entero	ID del destino
		-r	doble	Tasa de transmisión
		-D	doble	Retardo
		-h	doble	Longitud
		-O	orientación	orientación
		-b	caracteres	Etiqueta
		-c	color	Color
		-o	color	Color previo
		-S	caracteres	Estado (UP, DOWN)
		-l	caracteres	Etiqueta
		-L	caracteres	Etiqueta previa
		-e	color	Color de la etiqueta
		-E	color	Color previo de la etiqueta
Paquetes	h: Salto r: Recibe d: Línea caída	-t	tiempo	Tiempo
		-s	entero	ID de la fuente
		-d	entero	ID del destino

	+: En cola	-e	entero	Alcance
	-: Sale de cola	-a	entero	ID color atributo del paquete
		-i	entero	ID
		-l	entero	Energía
		-c	caracteres	Conversación
		-x	comentario	Comentario
		-p	caracteres	Tipo de paquete
		-k	caracteres	Tipo de paquete
		-y	comentario	
		-S	entero	
		-m	entero	
		-f	entero	
Sesión	E: En cola D: Sale de cola P: Caída	-t	tiempo	Tiempo
		-s	entero	ID de la fuente
		-d	entero	ID del destino
		-e	entero	Alcance
		-a	entero	Atributo
		-i	entero	ID
		-l	entero	Energía
		-c	caracteres	Conversación
		-x	comentario	Comentario
		-p	caracteres	Tipo de paquete
		-k	caracteres	Tipo de paquete
Agente	a	-t	tiempo	Tiempo
		-s	entero	ID de la fuente
		-d	entero	ID del destino
		-x	bandera	Quitar Agente
		-n	caracteres	Nombre de Agente
Característica	f	-t	tiempo	Tiempo
		-s	entero	ID de la fuente
		-d	entero	ID del destino
		-x	bandera	Quitar característica
		-T	caracter	Tipo
		-n	caracteres	Nombre
		-a	caracteres	Agente
		-v	caracteres	Valor
		-o	caracteres	Valor previo
Grupo	G	-t	tiempo	Tiempo
		-n	caracteres	Nombre
		-i	entero	ID del nodo
		-a	entero	ID del grupo
		-x	bandera	Quitar de grupo
Enlaces LAN	L	-t	tiempo	Tiempo
		-s	entero	ID de la fuente
		-d	entero	ID del destino
		-o	orientación	orientación
		-O	orientación	orientación
Marcas a nodos	m	-t	tiempo	Tiempo
		-n	caracteres	Nombre
		-s	entero	ID del nodo
		-c	caracteres	Color

		-h	caracteres	Forma (circle, square, hexagon)
		-X	bandera	Quitar marca
Evento de enrutamiento	R	-t	tiempo	Tiempo
		-s	entero	ID de la fuente
		-d	entero	ID del destino
		-g	entero	Grupo Multicast
		-p	paquete fuente	ID del paquete fuente ó *
		-n	bandera	Caché negativo
		-x	bandera	Ruta expirada
		-T	doble	Tiempo para expirar
		-m	caracteres	Modo (IIF ó OIF)
Rango inalámbrico	W	-t	tiempo	Tiempo
		-x	entero	X
		-y	entero	Y
Ejecutar expresión tcl	v	-t	tiempo	Tiempo
		-e	expresión tcl	expresión tcl
Jerarquía de configuración de direcciones -- solo en inicialización	A	-t	tiempo	Tiempo
		-n	entero	Jerarquía
		-p	entero	Puerto de turno
		-o	hexadecimal	Mascara del puerto
		-c	entero	Multicast de turno
		-a	entero	Mascara Multicast
		-h	entero	Jerarquía
		-m	entero	Nodo de turno
		-s	entero	Mascara del nodo
configuración de la tabla de colores -- solo inicialización	c	-t	tiempo	Tiempo
		-i	entero	ID
		-n	caracteres	Color
Creación de cola de paquetes -- solo inicialización	q	-t	tiempo	Tiempo
		-s	entero	ID de la fuente
		-d	entero	ID del destino
		-a	orientación	orientación
Esquema LAN	X	-t	tiempo	Tiempo
		-n	caracteres	Nombre
		-r	doble	Tasa
		-D	doble	Retardo
		-o	orientación	orientación
		-O	orientación	orientación

Formatos de los trazas del NAM.

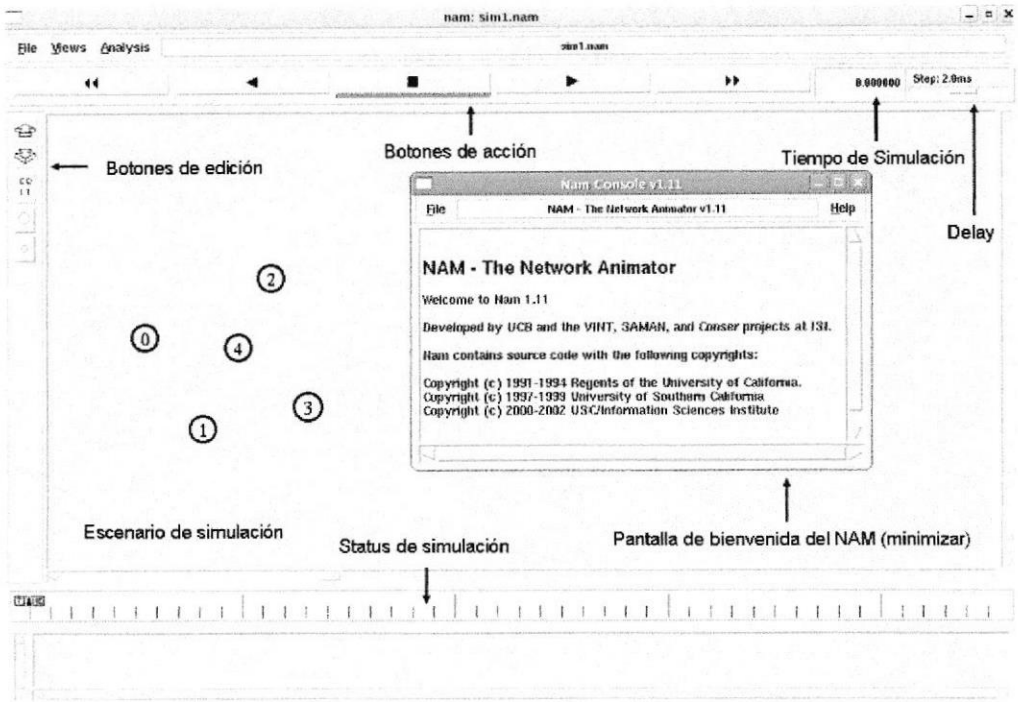
Para eventos en paquetes (entradas que empiezan con "h", "r", "d", "+", or "-"), los campos después de "-x"tienen el siguiente formato:

Evento	Tipo	Valor
Traza de nodo	{%s.%s %s.%s %d %s %s}	
	caracteres	Dirección del nodo fuente
	caracteres	Puerto del nodo fuente
	caracteres	Dirección del nodo destino
	caracteres	Puerto del nodo destino
	entero	Numero de secuencia

	caracteres	Banderas
	caracteres	Nombre del paquete

Formatos de las trazas adicionales del NAM.

Toda la información contenida en el archivo *.nam* se traduce en la ejecución de una simulación de forma grafica en la ventana del NAM, en intervalos de tiempos ajustables y con opciones novedosas parecidas a la de un reproductor para poder ver y manipular dicha simulación. En la siguiente figura podemos apreciar la pantalla principal del NAM para una simulación en particular en la cual se detallan las partes principales que la conforman.



Pantalla del NAM.

Dentro de NS2 se han creado varias librerías para poder simular escenarios específicos que no están contemplados por el simulador como lo son las simulaciones de redes Bluetooth para lo cual nos valemos de la librería UCBT de la Universidad de Cincinnati, la cual nos da los mejores resultados para la simulación de este tipo de redes. Sin embargo, para lograr que trabaje con el NAM desarrollamos una adaptación de esta librería en su versión 0.9.9.1. A continuación, se explican los comandos necesarios dentro del script para utilizar UCBT:

set val (mac) Mac/BNEP
esta sentencia indica que los nodos de la simulación trabajarán con la especificación Bluetooth.

set** variable_nam **[open** nombre_archivo.**nam w]
esta sentencia crea un archivo de texto con extensión NAM disponible para la escritura.

$**variable_simulador **namtrace-all-wireless $**variable_nam **x y
esta sentencia define que el escenario será de tipo wireless y con las dimensiones de las variables "x" y "y", además enlaza esta variable a la instancia del simulador.

Simulator set MacTrace_ ON
Simulator set RouterTrace_ ON
estas sentencias habilitan la escritura de paquetes de tipo MAC y de enrutamiento en el archivo de extensión NAM.

set node(*id_nodo****) [$****variable_simulador*** ***node*** *id_nodo*]
esta sentencia especifica como se crea un nodo en el NS2.

$node(*id_nodo****) rt*** *tipo_enrutamiento*
esta sentencia indica el tipo de enrutamiento usado en la simulación, sea este estático (ManualRT) o dinámico (AODV)

[$node(*id_nodo****) set l2cap_] set ifq_limit_*** *tamaño_cola*
esta sentencia especifica el tamaño máximo del número de paquetes L2CAP que puede atender cada nodo.

$node(*id_nodo****) inqscan*** *4096 2048*
$node(*id_nodo****) pagescan*** *4096 2048*
estas sentencias especifican los tiempos promedio que un nodo debe permanecer en los procedimientos de inquiry y paging.

$node(*id_nodo****) pos x y***
esta sentencia indica las coordenadas del nodo en el escenario de simulación.

Los tipos de transferencia de paquetes que soporta la librería UCBT son TCP y UDP, la definición de estos la detallamos a continuación:

set *var_tcp* ***[new Agent/TCP]***
$variable_simulador ***attach-agent $node(****tx_node****) $****var_tcp***
estas sentencias crean un agente TCP y se lo añade con un nodo que será el transmisor en una transferencia TCP.

set *var_ftp* ***[new Application/FTP]***
$var_ftp ***attach-agent $****var_tcp*
estas sentencias crean una aplicación FTP sobre el enlace TCP.

set *var_ack* ***[new Agent/TCPSink]***
$variable_simulador ***attach-agent $node(****rx_node****) $****var_ack*
estas sentencias establecen el nodo que recibirá los paquetes TCP y que a su vez enviará los paquetes ACK al nodo transmisor.

$variable_simulador ***connect $****var_tcp* ***$****var_ack*
esta sentencia realiza la conexión entre los nodos trasmisor y receptor.

set *var_queue* ***[new Queue/DropTail]***
$var_queue ***set limit_ 20***
estas sentencias definen e indica el número de paquetes que pueden estar almacenados en la cola de espera.

Sentencias utilizadas para la transferencia de paquetes UDP:

set** var_udp **[new Agent/UDP]
***$**variable_simulador **attach-agent $node(**tx_node**) $**var_udp*
estas sentencias crean un agente UDP y se lo añade con un nodo que será el transmisor en una transferencia UDP.

set** var_cbr **[new Application/Traffic/CBR]
***$**var_cbr **attach-agent $**var_udp*
estas sentencias crean una aplicación CBR sobre el enlace UDP.

set** var_null **[new Agent/Null]
***$**variable_simulador **attach-agent $node(**rx_node**) $**var_null*
estas sentencias establecen el nodo que recibirá los paquetes UDP.

***$**variable_simulador **connect $**var_udp **$**var_null*
esta sentencia realiza la conexión entre los nodos transmisor y receptor.

Para la organización de los eventos se utilizan los siguientes comandos:

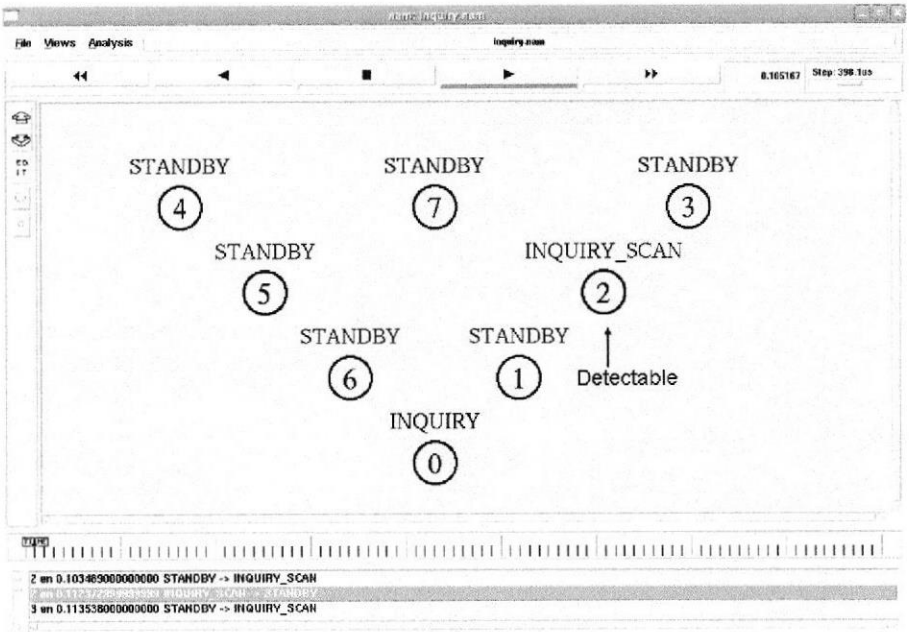
***$**variable_simulacion **at** t1 **"$node(**master**) make-bnep-connection $node(**slave**)** pack_master pack_slave var_queue*
esta sentencia establece la conexión maestro - esclavo y se indica el tipo de paquetes, en el caso de paquetes UDP el *var_queue* es reemplazado por *NONE*. Esta línea se ejecutara en el tiempo correspondiente a *t1*.

***$**variable_simulacion **at** t2 **"$node(**master**) make-br $node(**bridge**)** pack_master pack_bridge var_queue*
esta sentencia establece una conexión entre ambos nodos, especificando que el nodo "*bridge*" podrá aceptar múltiples conexiones activas, en el caso de paquetes UDP el *var_queue* es reemplazado por *NONE*.

Una vez que terminamos de definir todos los eventos de la simulación se debe llamar al procedimiento de finalización.

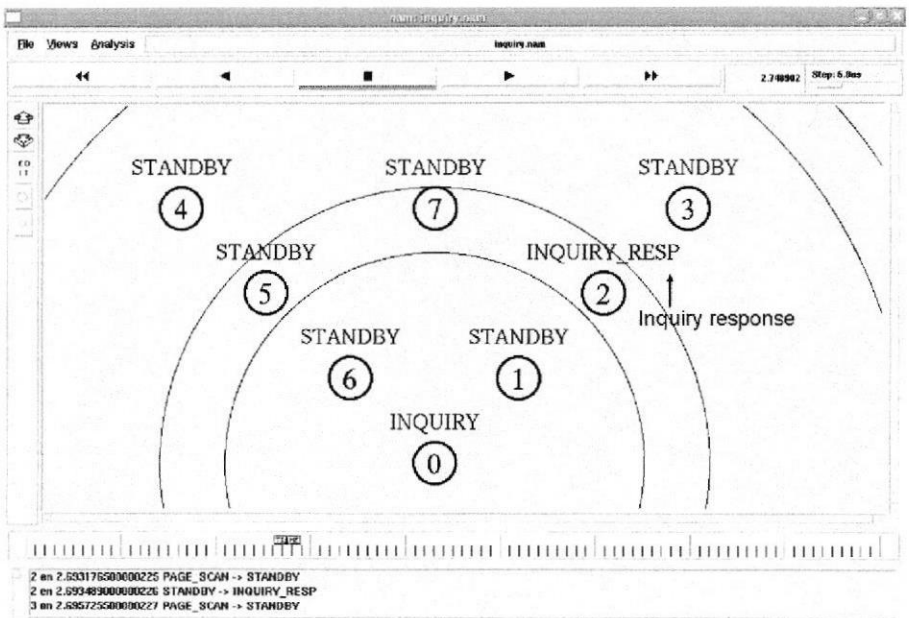
$** variable_simulacion **at** t3 **"finish"
esta sentencia invoca en el tiempo *t3*, el cual hace referencia a la finalización de la simulación, al procedimiento de finalización *"finish"*.

En las siguientes figuras se muestran los procedimientos básicos de redes Bluetooth utilizando las adaptaciones realizadas a la librería UCBT. Primero encontramos el procedimiento de detección (descubrimiento) de nodos dentro del área de cobertura a lo que se conoce con el nombre de *inquiry*.



Nodo 0 en procedimiento de Inquiry.

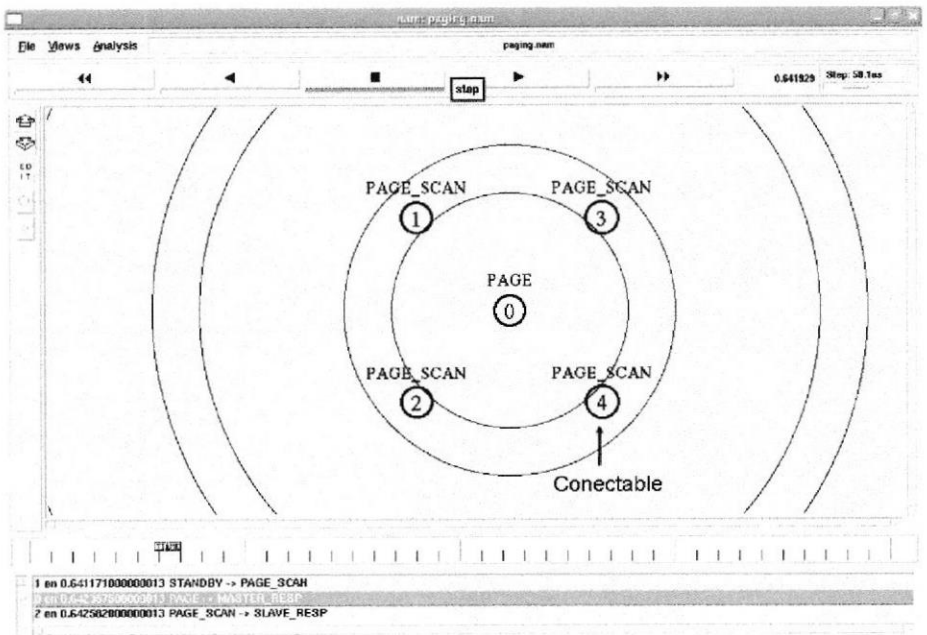
Podemos apreciar a varios dispositivos en una formación en "V" donde los dispositivos que se encuentren en el estado de INQUIRY_SCAN son aquellos que podrán ser detectados por el dispositivo que se encuentra en el estado de INQUIRY.



Nodo 2 envía una respuesta al procedimiento de Inquiry del Nodo 0.

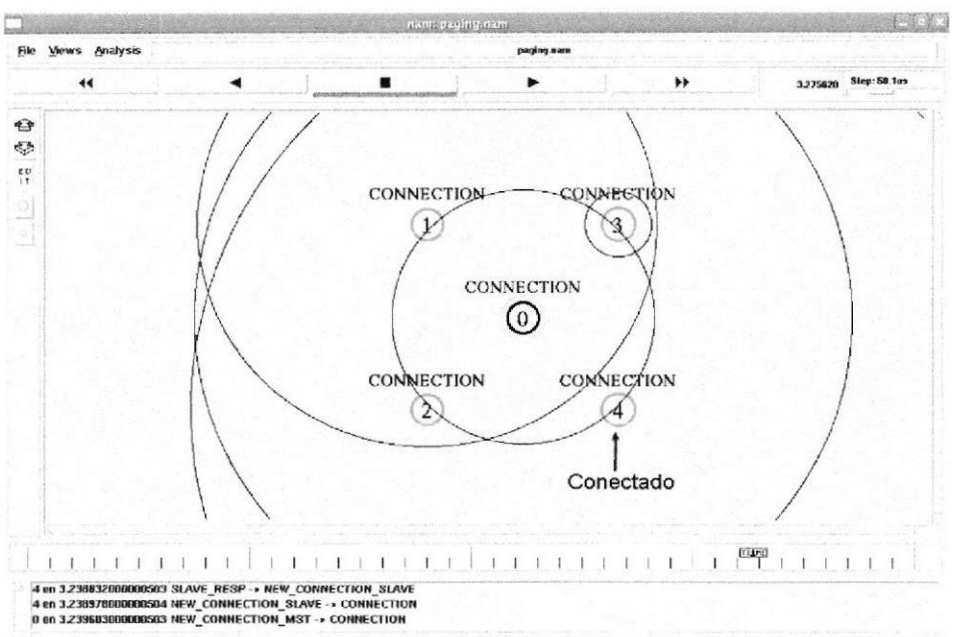
Una vez que el dispositivo que está buscando a otros (el que se encuentra en INQUIRY), envía peticiones a los dispositivos que pueden ser detectados (en INQUIRY_SCAN), estos envían una respuesta para que sean registrados y se confirme que están dentro del área de cobertura para poder luego conectarse entre sí. Cuando se han detectado los dispositivos que estarán próximos a ser interconectados, se realiza el procedimiento de conexión (Paging), Dentro de este procedimiento los dispositivos

pueden estar en 2 estados, el de PAGE_SCAN (conectable) y el de PAGE (buscando conectar dispositivos).



Nodo 0 en procedimiento de Page.

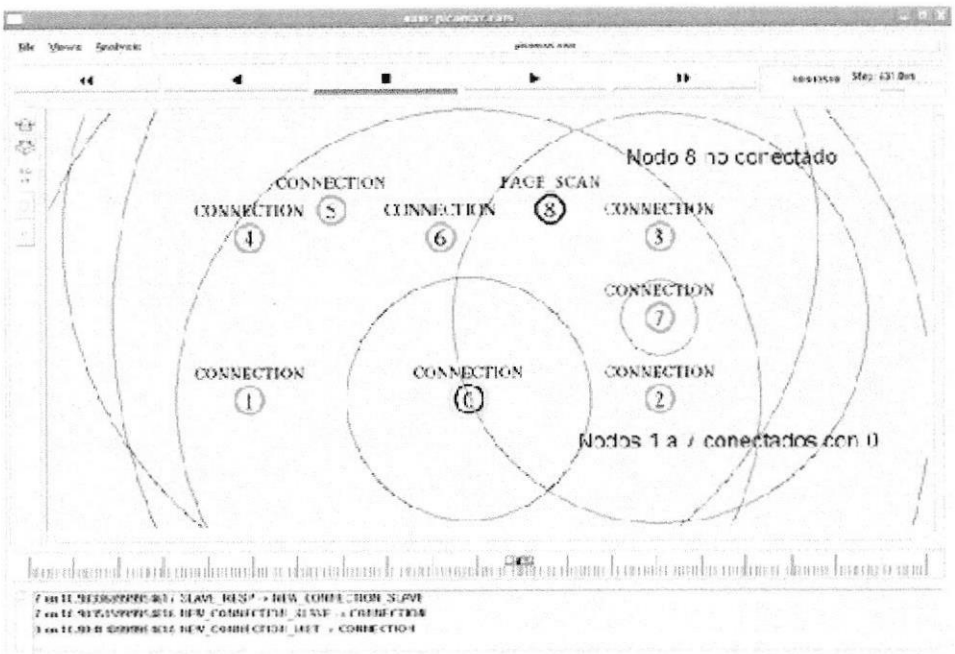
Para que los dispositivos entren en el estado de CONNECTION (conectado) deben primero realizar un par de peticiones y sincronizaciones entre los dispositivos que están participando, una vez que se efectué la respuesta del PAGE, se empezara a intercambiar comandos LMP para establecer una piconet (forma básica de conexión en dispositivos Bluetooth).



Nodo 1, 2, 3, 4 conectados al Nodo 0.

Podemos apreciar como los nodos que intercambiaron las respuestas de PAGE y los comandos LMP necesarios para establecer la conexión son aquellos que tienen coloración verde y con el estado CONNECTION (conectado), cabe recalcar que en este

procedimiento se designan roles entre los dispositivos, el dispositivo que inicio con el PAGE es aquel que tendrá el role de MASTER (maestro, el dispositivo que podrá interconectar a más nodos dentro de la misma red), y aquellos que enviaron la respuesta tendrán el role de SLAVES (esclavos, nodos sujetos al esquema que imponga el dispositivo maestro).



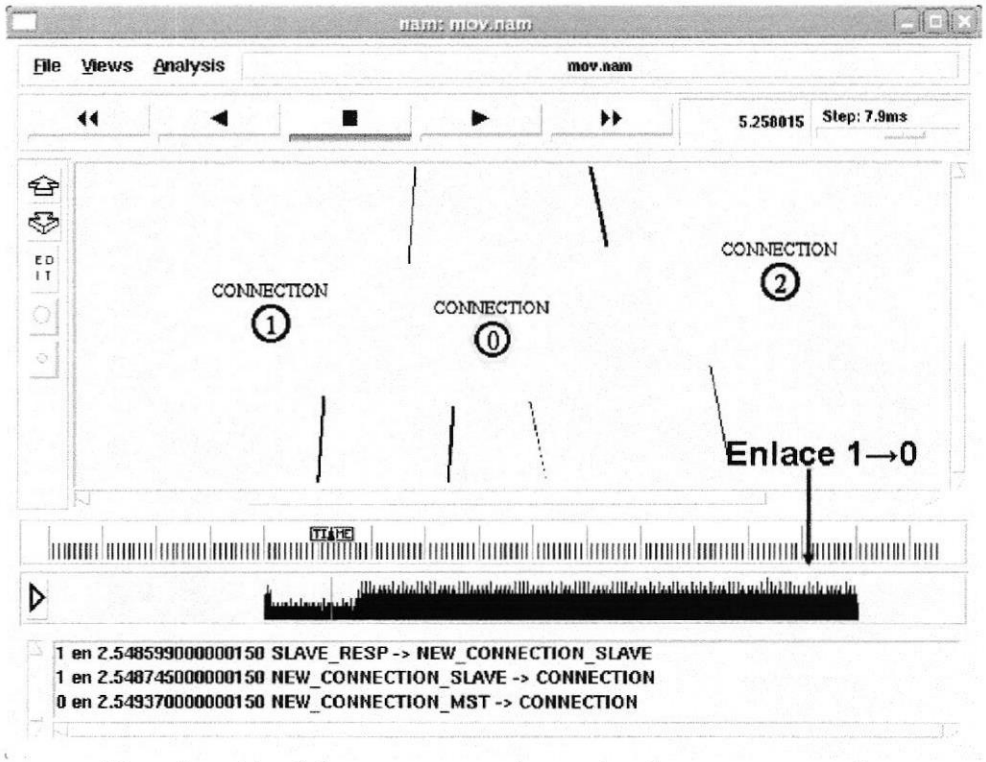
Formación de una piconet.

En la figura anterior tenemos una piconet con 8 nodos conectados, los nodos del 1 al 7 conectados con al nodo 0 que es el maestro de esta piconet (por tal motivo lleva una marca circular amarilla), y vemos que el nodo 8 no se encuentra conectado pero fue detectado y es conectable puesto que está en el estado de PAGE_SCAN, este nodo no se logra conectar debido a que la especificación Bluetooth define que una piconet puede tener hasta 7 esclavos activos conectados a un maestro.

En este momento los nodos conectados al maestro están listos para efectuar transferencias de paquetes entre esclavos o entre esclavos y el maestro en caso de que se hubiese detallado eso en la simulación.

Una de las herramientas que nos brinda el NAM y la cual podemos utilizar durante el tiempo que transcurra la simulación es la visualización de la ocupación de los enlaces entre los nodos que se encuentren conectados. Esta opción se puede activar dando click derecho a un costado de uno de los nodos que interviene en la transferencia de paquetes en dirección hacia el otro nodo conectado.

Al realizar esta acción nos saldrá un menú contextual de donde seleccionamos la opción "Graph" y por último seleccionamos el enlace que nos interese visualizar ya que nos permite obtener gráficos de los enlaces en ambos sentidos. La siguiente figura nos muestra la activación de esta opción en la ventana del NAM y nos muestra la ocupación del enlace entre el nodo 0 y el nodo 1 en el sentido del nodo 1 hacia el nodo 0.



Visualización del momento en el que el enlace esta ocupado.

APÉNDICE B.

Prácticas Didácticas de Simulación de Redes Bluetooth.

Estos apéndices de prácticas didácticas tienen el propósito de guiar al estudiante al desarrollo de las simulaciones de redes Bluetooth. Antes de empezar a trabajar con estas prácticas el estudiante debe estar familiarizado con los tutoriales de NS2, NAM y UCBT, descritos en el apéndice A.

B.1 Práctica No. 1 "Simulación de procedimientos básicos para la conexión de dispositivos Bluetooth basados en NS2/UCBT/NAM"

Práctica No.1

Simulación de procedimientos básicos para la conexión de dispositivos Bluetooth basados en NS2/UCBT/NAM

1. OBJETIVOS

- Orientar a los estudiantes al manejo de las simulaciones de redes Bluetooth utilizando el simulador de redes NS2.
- Conocer y visualizar los procedimientos que están incluidos en la formación de redes Bluetooth.
- Crear una topología de red Bluetooth básica (Piconet).
- Verificar el número máximo de enlaces activos dentro de una piconet.

2. INTRODUCCION

En esta práctica se aprenderán varios procedimientos básicos para establecer una comunicación vía Bluetooth, dentro de los cuales se mencionan: el Inquiry o Descubrimiento, el Paging o Emparejamiento, y el intercambio de paquetes necesarios para poder establecer una comunicación entre dispositivos.

UCBT es una librería compatible con el simulador de redes NS2. El objetivo de esta librería es utilizar los nodos creados en NS2 y adaptarlos para que funcionen dentro de la especificación Bluetooth, en lo que se

refiere a conexión, transferencia de archivos, rango de cobertura, movilidad, entre otros.

3. MARCO TEORICO

Bluetooth es una norma que define un standard global de comunicación inalámbrica, que posibilita la transmisión de voz y datos entre diferentes equipos mediante un enlace por radiofrecuencia. Los dispositivos Bluetooth operan en una frecuencia de radio que se encuentra entre los 2.4 y 2.48GHz, con la posibilidad de realizar hasta 1.600 saltos por segundo entre las 79 frecuencias soportadas en intervalos de 1MHz. Cada unidad incluye una radio, un controlador de enlaces de banda base y el software para la administración de los enlaces y flujo de datos.

Los usuarios tienen la opción de dos potencias de señal: un nivel de baja potencia para distancias de hasta 10 metros, y un nivel de alta potencia de hasta 100 metros de distancia para los puntos de acceso. Los dispositivos Bluetooth pueden conectarse simultáneamente hasta siete aparatos más, sin incluir el dispositivo maestro. La velocidad máxima de transferencia de datos es de aproximadamente 720 Kbps por canal.

4. MATERIALES Y EQUIPOS

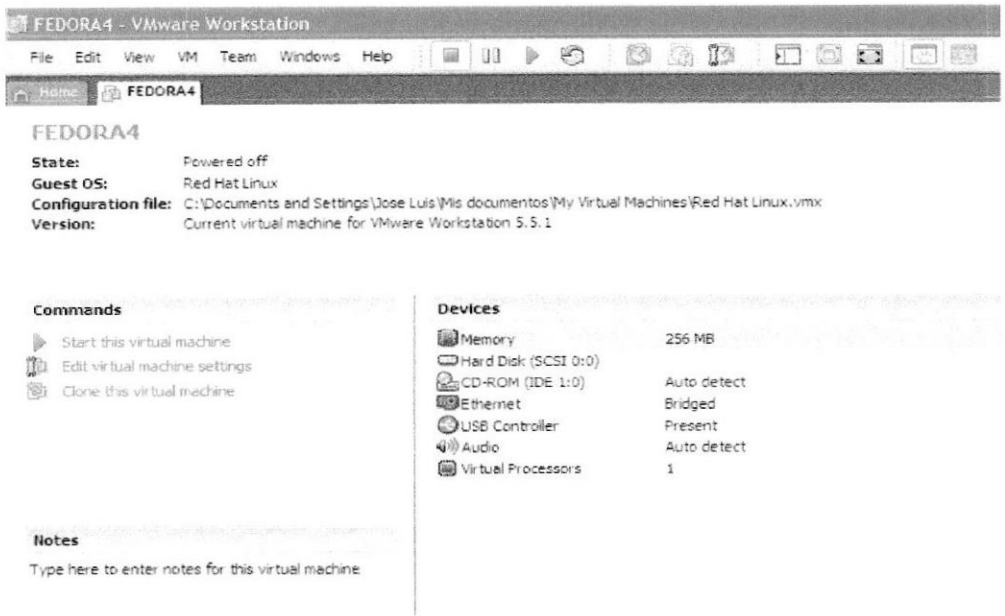
- PC con Fedora Core 4 y Network Simulator 2: Network Animator, librería UCBT modificada.

5. PROCEDIMIENTOS

Abriremos el emulador de sistemas operativos "VMware Workstation", para esto haga doble click en el icono **VMware Workstation 5.5.1** en el escritorio o vaya a la siguiente dirección:

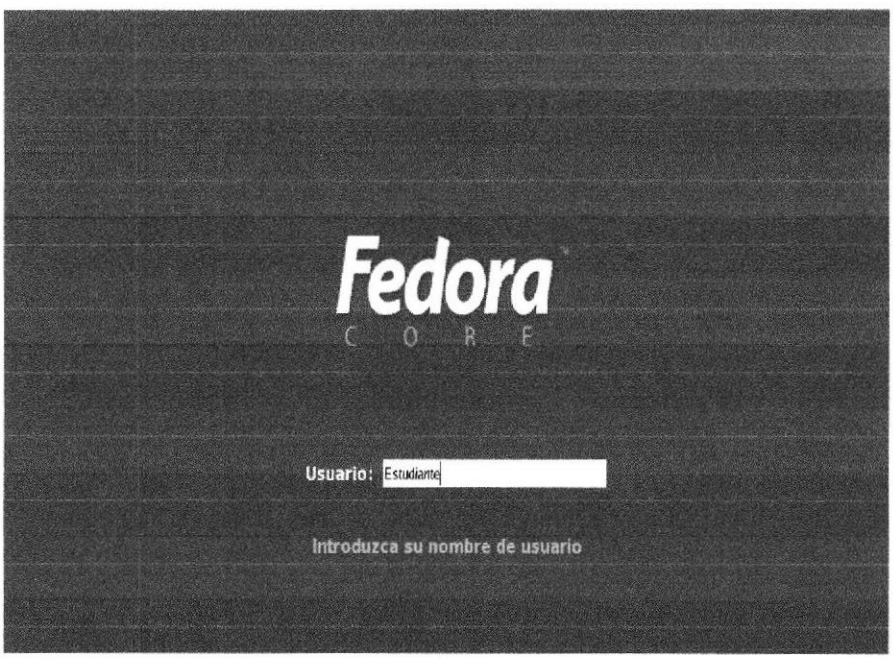
Inicio->Programas->VMware->VMware Workstation 5.5.1

aparecerá la siguiente ventana:

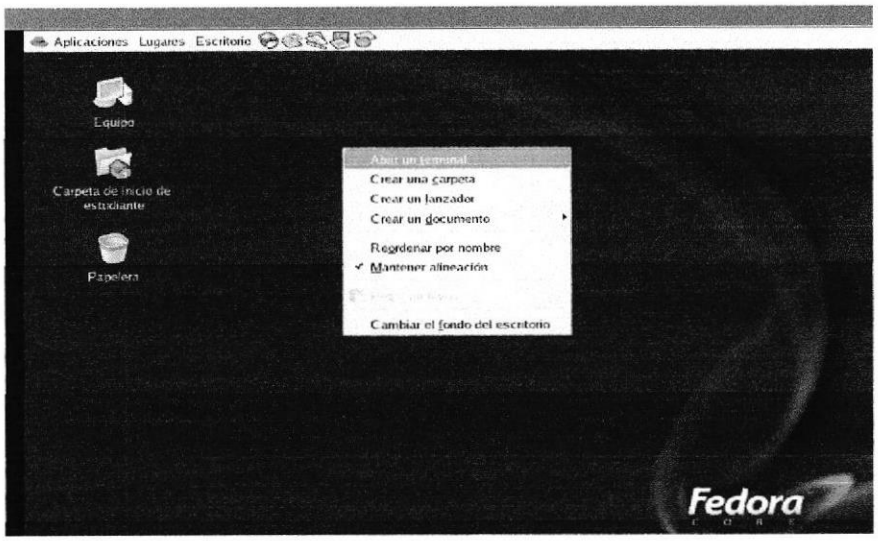


desde aquí haga click en "*Start this virtual machine*" para abrir el sistema operativo "Fedora Core 4".

En la pantalla de inicio de sesión de Fedora Core 4 proceda a digitar en el campo usuario lo siguiente: Estudiante, como se muestra a continuación:



Luego, en el campo contraseña, proceda a digitar lo siguiente: Estudiante, y al finalizar presione "*Enter*". Dentro del escritorio abrimos una ventana de terminal dando click derecho sobre el escritorio.

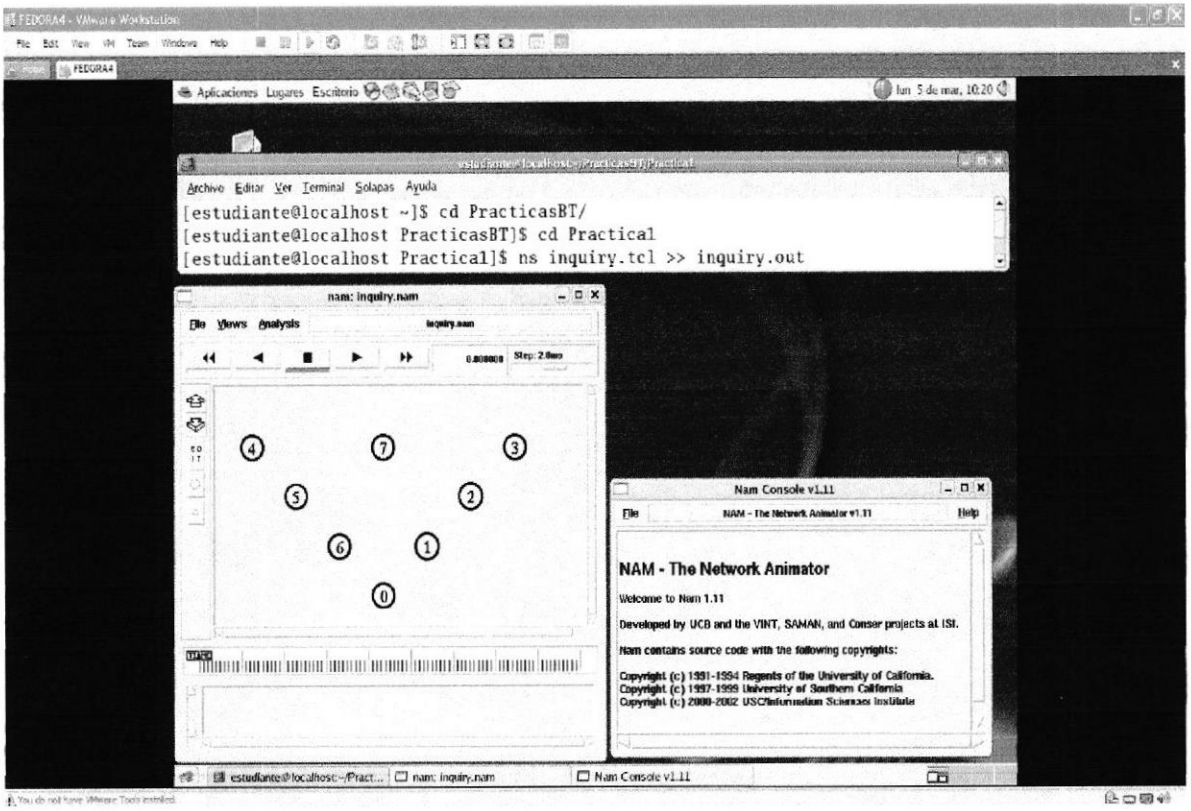


5.1 Simular un procedimiento de Inquiry y observar los cambios de estados de los nodos.

El procedimiento de Inquiry permite a un dispositivo ser descubierto por otros dispositivos que se encuentren en su rango y a su vez determina las direcciones y sus respectivos clocks. Se involucra un dispositivo (la fuente) el cual envía paquetes de inquiry (estado inquiry) y después recibe la respuesta correspondiente. El dispositivo que recibe los paquetes de inquiry (el destino) debe encontrarse en el sub-estado de inquiry scan para poder recibir dichos paquetes. Después el dispositivo destino entrará al sub-estado de inquiry response y enviará una respuesta de inquiry a la fuente. Luego que el procedimiento de inquiry se ha completado, se establece la conexión mediante el procedimiento de Paging.

Una vez que tengamos la pantalla del terminal, ejecutaremos las siguientes sentencias:

- cd PracticasBT
- cd Practica1
- ns inquiry.tcl >> inquiry.out



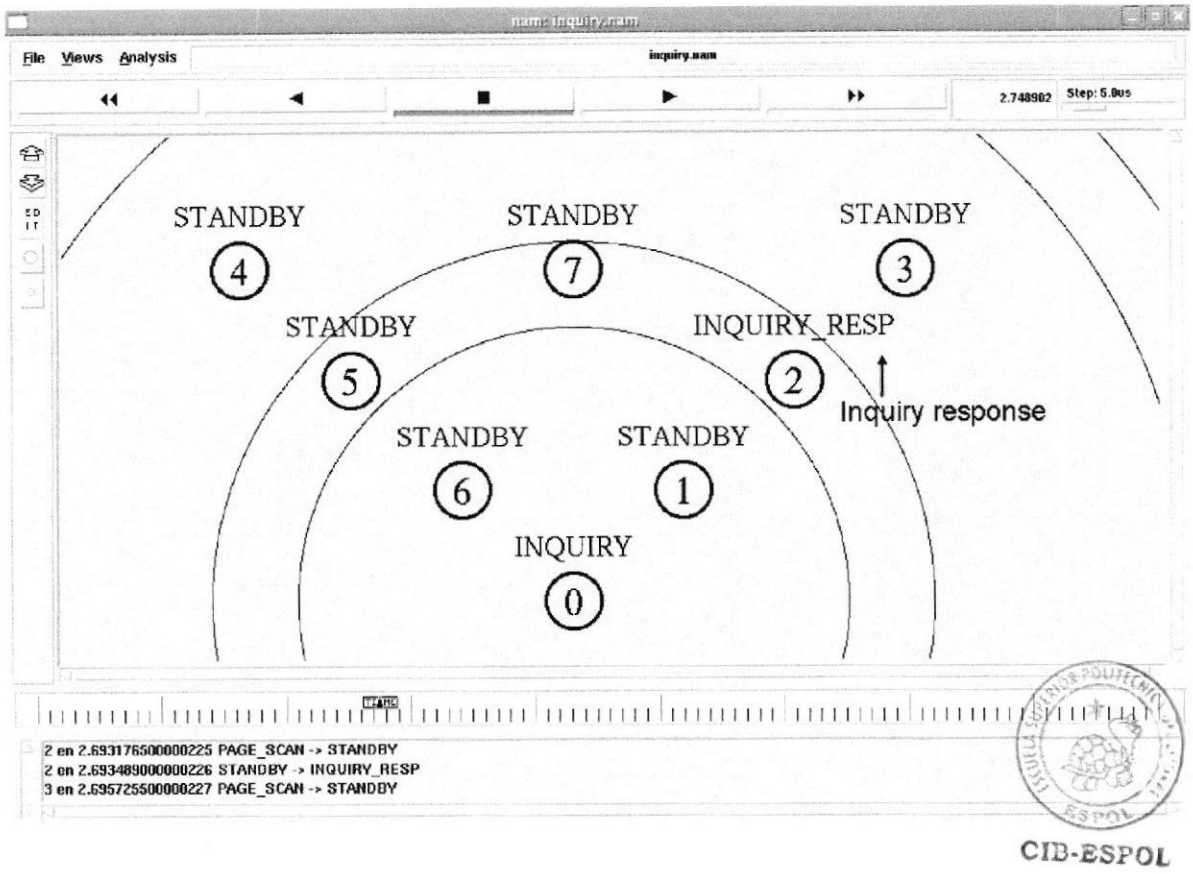
Al presionar *play* en la presente simulación se ejecutarán los estados de descubrimiento de dispositivos en los nodos 0 y 7, durante un tiempo determinado. El comando que se utilizará es el siguiente:

$ns at 0.1 "$node(0) inquiry 5 5"

Lo que implica que el nodo 0 permanecerá en un estado de inquiry durante 5x1.28seg. a menos que reciba 5 inquiry responses de dispositivos diferentes. Para el nodo 7 se ejecutará la siguiente sentencia:

$ns at 6.0 "$node(7) inquiry 2 5"

En la siguiente figura se puede apreciar el momento en el cual el nodo 2 envía un inquiry response hacia el nodo 0.



CIB-ESPOL

Al finalizar esta simulación conteste las siguientes preguntas:

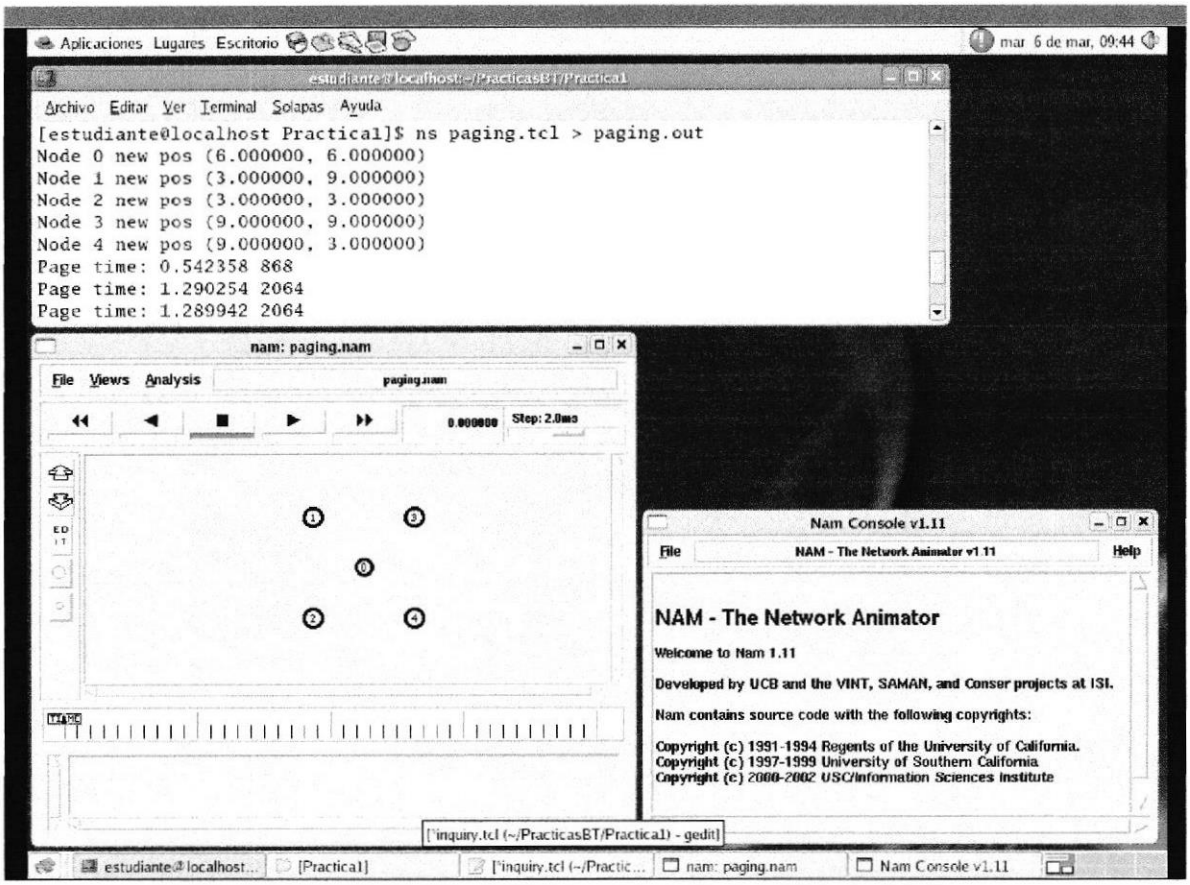
- ¿Cuáles son los 5 nodos que envían *"inquiry responses"* al nodo 0 y en que tiempo ocurre cada respuesta?
- ¿Cuánto tiempo se toma el nodo 0 durante el proceso de inquiry?
- ¿En que tiempo termina el nodo 7 su procedimiento de Inquiry?
- ¿Cuáles son los nodos que envían *"inquiry response"* al nodo 7?
- ¿Cuanto tiempo se toma el nodo 7 durante el proceso de Inquiry?

5.2 Simular un procedimiento de Paging y observar los roles que adquiere los nodos.

Una vez que conocemos los nodos disponibles para conectarse, se realiza el procedimiento de conexión o Paging, donde los nodos intercambiarán información con respecto a su posición y su frecuencia de reloj CLK para proceder a sincronizarse a un esquema dominante. El nodo que se impone en este esquema se denominará "maestro" y los que lo seguirán se llamarán "esclavos".

En la ventana del terminal proceda a ejecutar la siguiente sentencia:

- ns paging.tcl >> paging.out

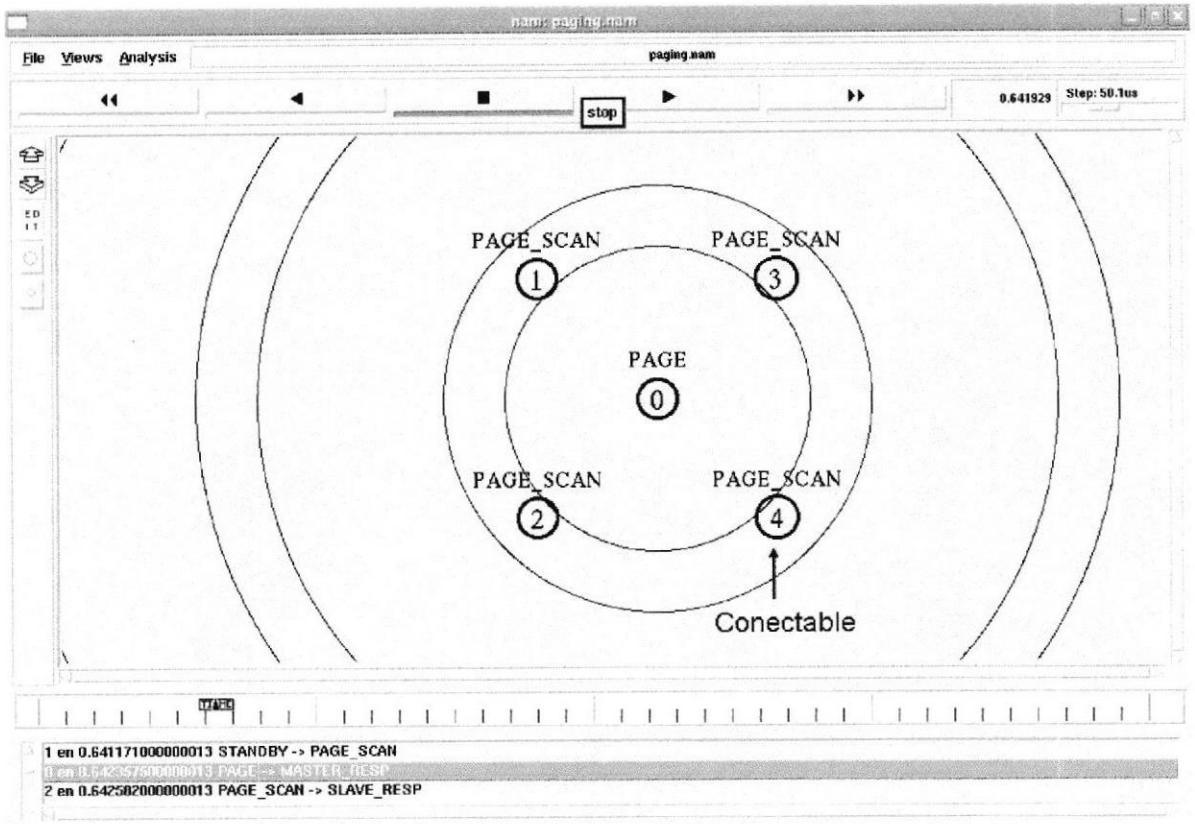


Al presionar el botón *play* en la presente simulación, podemos observar que se realizara el procedimiento de conexión de dispositivos, paging, dando como resultado un dispositivo maestro y varios dispositivos esclavos. Esto se lo consigue con el siguiente comando:

$ns at 0.1 "$node(0) make-bnep-connection $node(1)"

Este comando permite establecer una conexión hasta la capa BNEP dentro de la pila de protocolos Bluetooth entre el nodo 0 y el nodo 1. Ambos dispositivos entrarán en el procedimiento de paging y de page scan respectivamente, realizando intercambios de paquetes necesarios para la conexión de ambos dispositivos.

En la figura se pueden apreciar los estados de *page* y *page scan* respectivamente:



Al finalizar esta simulación conteste las siguientes preguntas:

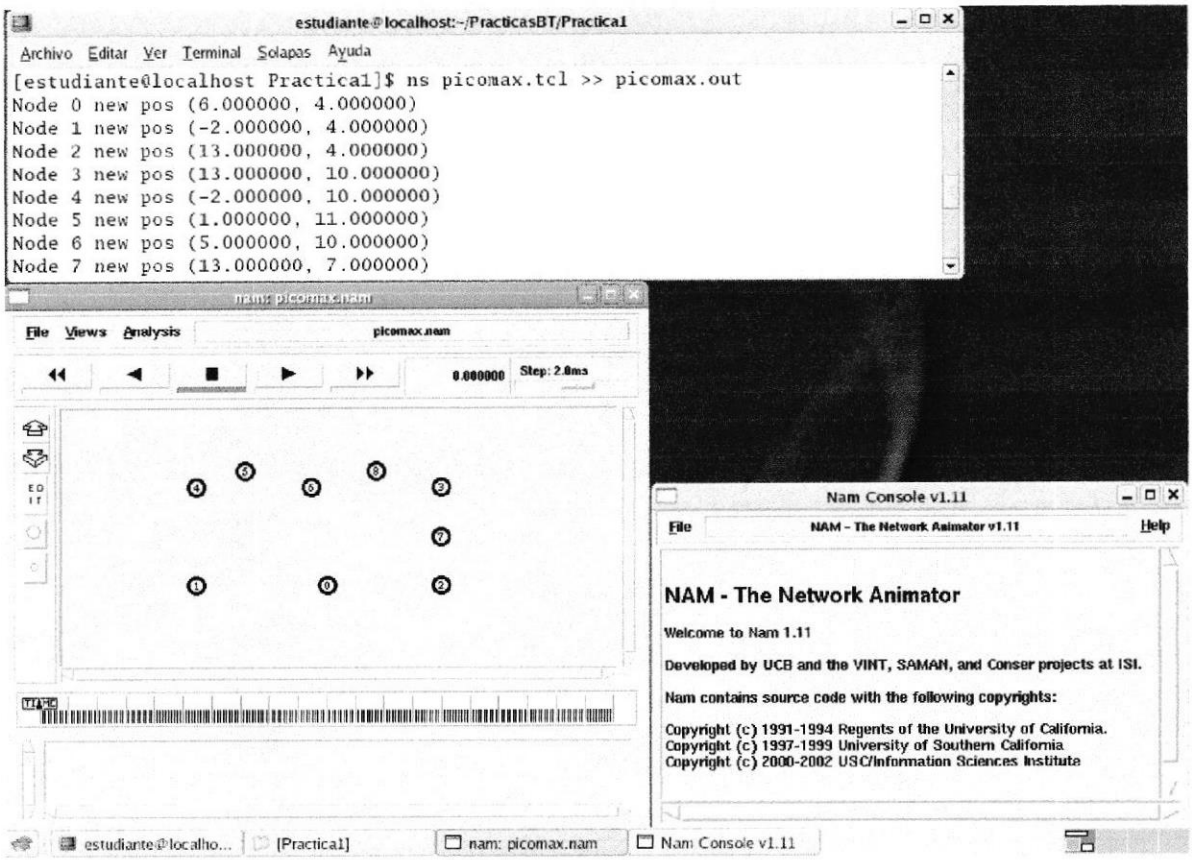
- ¿Qué tipo de paquetes envía y recibe el nodo 0?
- ¿Qué tipo de paquetes envían y reciben los otros nodos?
- ¿Qué nodos logran el rol de maestro?
- ¿Qué nodos logran el rol de esclavo?
- ¿Por cuáles estados deben de pasar los nodos hasta llegar al estado de conexión? Haga una tabla con cada nodo.

5.3 Simular una piconet con la cantidad máxima de nodos conectados y con una transferencia de archivos bajo una aplicación TCP.

Una vez concluido el procedimiento de paging, los dispositivos tomaron roles específicos dentro de una conexión común llamada piconet. Una piconet consta de un único dispositivo maestro al cual se conectan varios dispositivos esclavos. El número máximo de dispositivo esclavos "activos" dentro de una piconet es de 7. Una vez conectados, los nodos pueden intercambiar archivos según aplicaciones que se demanden entre dispositivos. Estas aplicaciones pueden ser basadas en el protocolo TCP o UDP según la aplicación.

En la ventana del terminal proceda a ejecutar la siguiente sentencia:

- ns picomax.tcl >> picomax.out

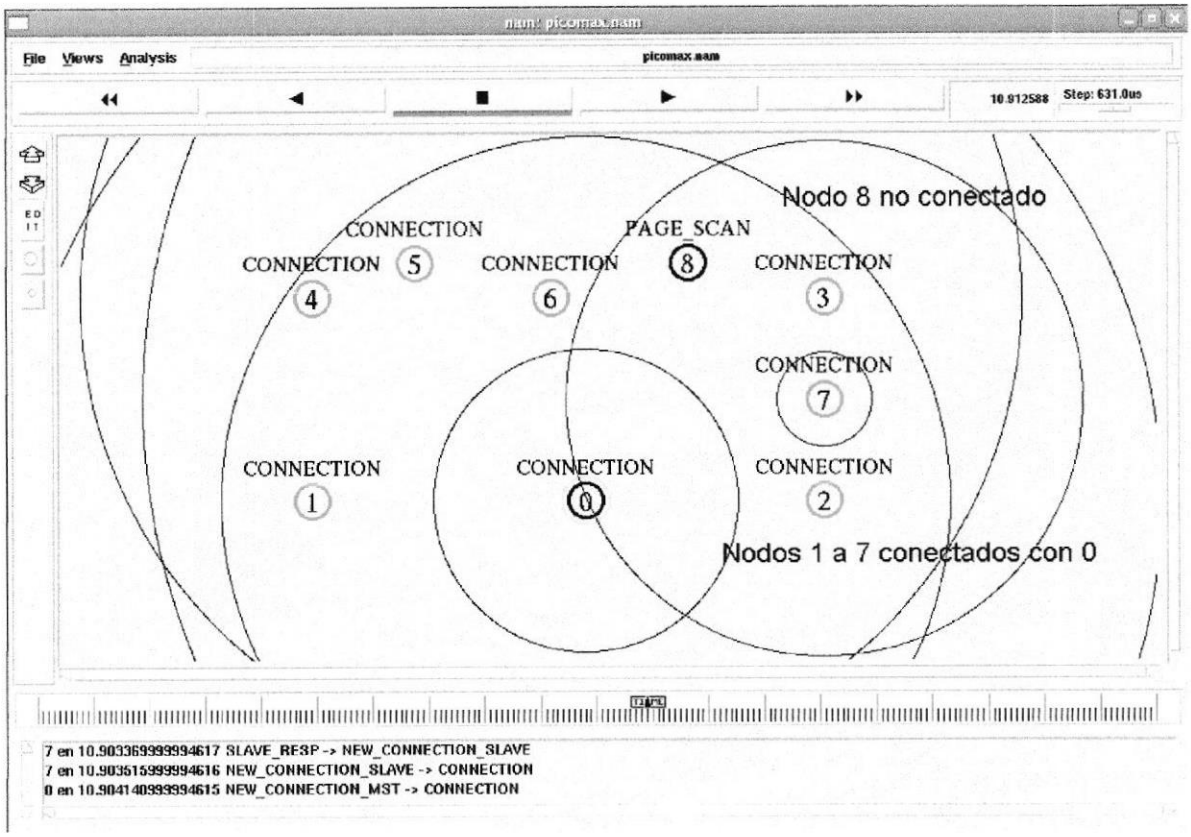


Al presionar el botón *play* en la presente simulación, podemos observar la formación de una piconet con el número máximo de nodos esclavos activos (7). Además habrá una transferencia de tipo TCP entre los nodos 1 y 2 que se la consigue con las siguientes líneas de código:

$ns at 0.1 "$node(0) make-bnep-connection $node(1) DH3 DH3 noqos $ifq"
$ns at 0.2 "$node(0) make-bnep-connection $node(2)"
$ns at 12 "$ftp0 start"

Estas líneas de código permiten establecer una aplicación de tipo FTP entre los nodos 1 y 2 mediante una transferencia de archivos de tipo TCP. Antes de empezar el traspaso de datos, se debe realizar una conexión hasta la capa BNEP entre el nodo maestro 0 y los nodos esclavos 1 y 2, respectivamente.

En la siguiente figura podemos apreciar la formación de una piconet:



Al finalizar esta simulación conteste las siguientes preguntas:

- Explique la función del nodo con marca amarilla y los nodos con marca verde.
- Explique que sucede con el nodo 8.
- En la transferencia de paquetes entre los nodos 1 y 2, explique por qué el tráfico pasa por el nodo 0.

BIBLIOGRAFIA

1. The VINT Project, The ns Manual, UC Berkeley, LBL, USC/ISI, and Xerox PARC,.Kevin Fall (kfall@ee.lbl.gov), Kannan Varadhan kannan@catarina.usc.edu), Editores; Enero 8, 2003.

2. BluetoothSIG, Specifications Documents, http://www.bluetooth.com/Bluetooth/Learn/Technology/Specifications/ revisada el 4 de Enero 2006.

3. D. Agrawal, Q. Wang, UCBT Bluetooth Extension for NS2 at the University of Cincinnati, http://www.ececs.uc.edu/~cdmc/ucbt/ucbt.html revisada el 15 de Enero de 2006.

B.2 Práctica No. 2 "Transferencia de paquetes en diferentes topologías de redes Bluetooth basados en NS2/UCBT/NAM"

Práctica No.2

Transferencia de paquetes en diferentes topologías de redes Bluetooth basados en NS2/UCBT/NAM

1. OBJETIVOS

- Examinar escenarios móviles de redes Bluetooth y verificar sus rangos de cobertura.
- Conocer el funcionamiento de los nodos PMPs y su importancia en las redes Bluetooth.
- Mostrar el intercambio de roles de dos dispositivos conectados a una piconet.
- Formar una topología de red Bluetooth avanzada (Scatternet).

2. INTRODUCCIÓN

En esta práctica se aprenderán escenarios avanzados de redes Bluetooth, donde tenemos topologías con gran número de nodos y varias aplicaciones simultáneas, así como el intercambio de archivos entre nodos bajo aplicaciones FTP y CBR y procedimientos de intercambio de roles entre dispositivos conectados.

En la librería UCBT se ha especificado como distancia máxima entre dispositivos Bluetooth cerca de 10 metros a la redonda, es decir la cobertura máxima de cada nodo será esa distancia. El objetivo es utilizar los nodos con las potencias mínimas, teniendo en consideración que por el hecho de ser dispositivos móviles, estos requieren un sistema de ahorro de energía.

3. MARCO TEORICO

Un dispositivo Bluetooth puede atender más de una conexión a la vez, y esto lo logra porque cuando se establece una piconet se usan ciertos canales en un esquema de salto de frecuencia, por lo que cada piconet tendrá un

esquema propio. Estos dispositivos que participan en más de una piconet a la vez se los conoce como nodos PMP (Participante en Múltiples Piconets), estos nodos suelen desempeñar tareas importantes cuando de interconectar piconets se trata, es decir, forman un enlace o puente para la interconexión entre piconets. A lo que se conoce con el nombre de scatternet.

Muchas veces dentro del establecimiento de una piconet o incluso de una scatternet, los dispositivos requieren hacer un cambio de roles para poder mejorar su topología y establecer una conexión con la menor cantidad de problemas. A este procedimiento se lo conoce con el nombre de Role-Switch.

Un nodo puente puede tener roles de maestro o de esclavo, según como se realice la interconexión. Debido a esto y otros factores, las tasas de datos en este tipo de topología pueden disminuir según sea el nivel de interferencia o la cantidad de dispositivos y aplicaciones simultáneas que se atiendan. Lo bueno de este tipo de topología es que de alguna manera puede ampliar la cobertura entre dispositivos que se encuentren muy lejanos a los 10 metros.

4. MATERIALES Y EQUIPOS

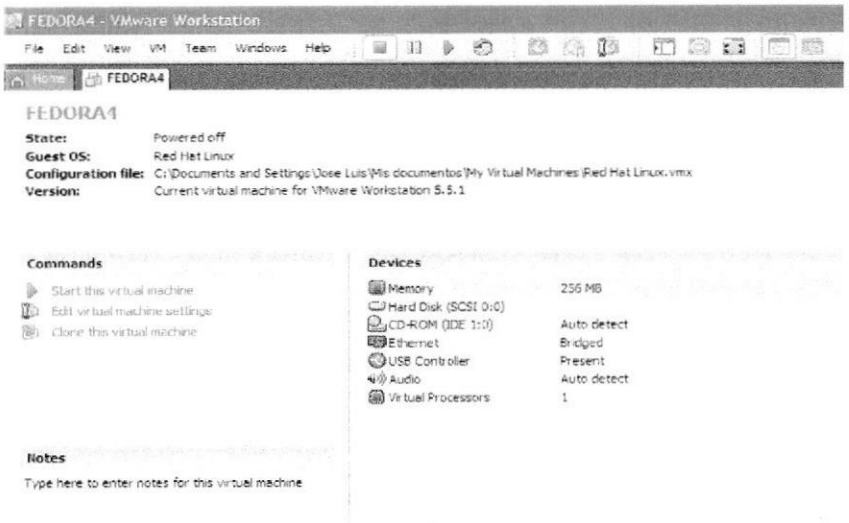
- PC con Fedora Core 4 y Network Simulator 2: Network Animator, librería UCBT modificada.

5. PROCEDIMIENTOS

Abriremos el emulador de sistemas operativos "VMware Workstation", para esto haga doble click en el icono **VMware Workstation 5.5.1** en el escritorio o vaya a la siguiente dirección:

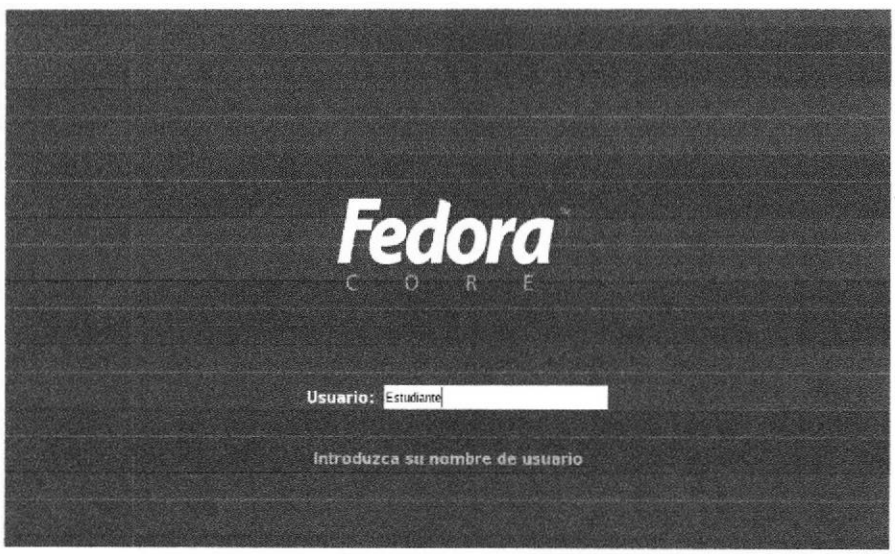
Inicio->Programas->VMware->VMware Workstation 5.5.1

aparecerá la siguiente ventana:

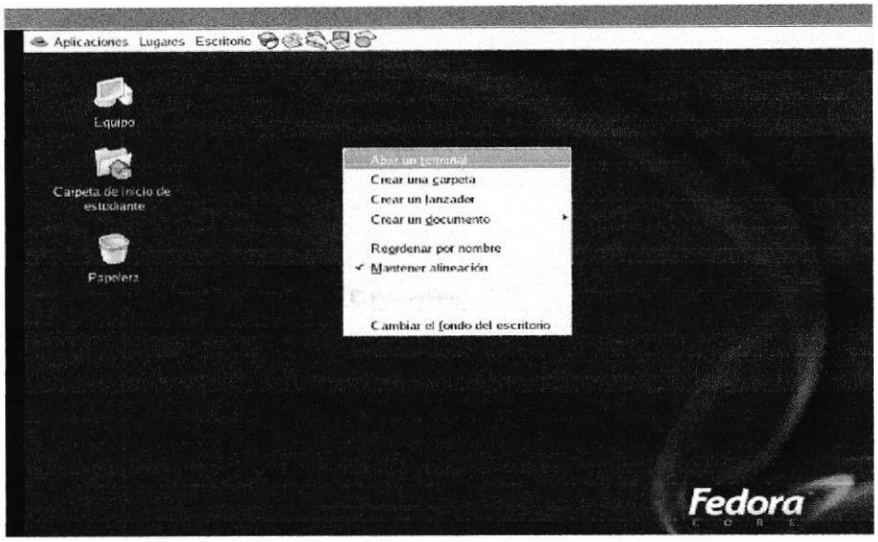


desde aquí haga click en "*Start this virtual machine*" para abrir el sistema operativo "Fedora Core 4".

En la pantalla de inicio de sesión de Fedora Core 4 proceda a digitar en el campo usuario lo siguiente: Estudiante, como se muestra a continuación:



Luego, en el campo contraseña, proceda a digitar lo siguiente: Estudiante, y al finalizar presione "*Enter*". Dentro del escritorio abrimos una ventana de terminal dando click derecho sobre el escritorio.

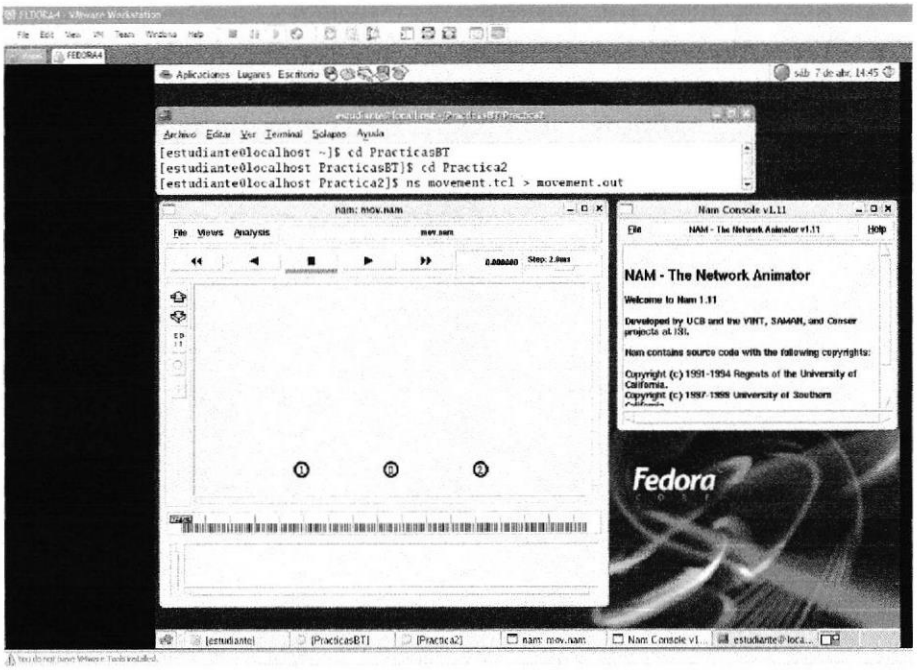


5.1 Simular una piconet con nodos móviles y analizar los rangos de cobertura.

En todo momento, los dispositivos conectados en una piconet deben estar dentro de un espacio físico en el cual puedan comunicarse, es decir, logren un intercambio efectivo de paquetes de información y control. A este espacio físico se lo denomina área de cobertura. Por definición, todos los nodos tienen un área de cobertura de 10 metros para potencias bajas, debido a esto, cuando los dispositivos han establecido una piconet significa que entre el dispositivo maestro y los dispositivos esclavos existen como máximo una distancia de 10 metros.

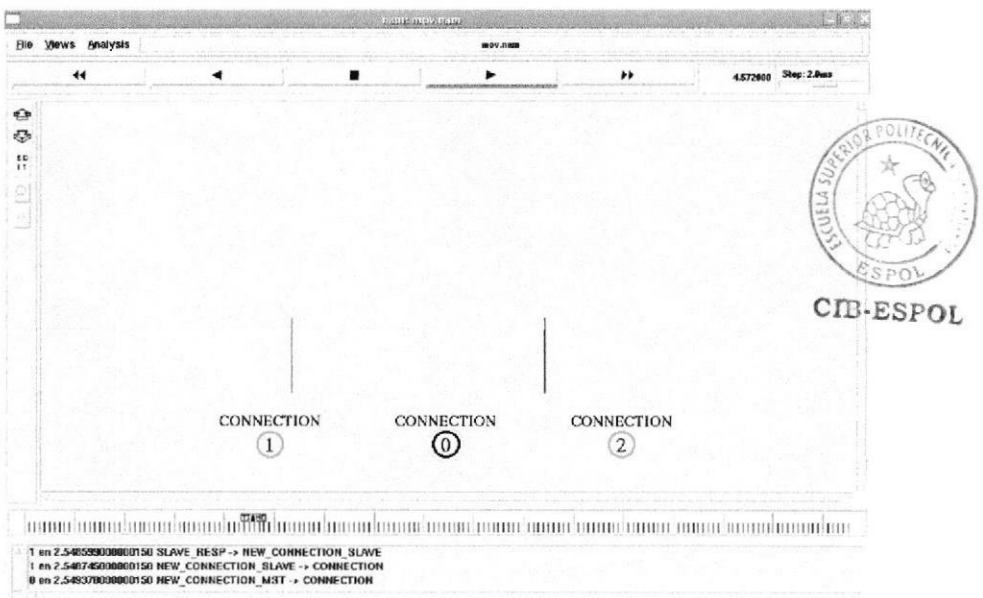
Una vez que tengamos la pantalla del terminal, ejecutaremos las siguientes sentencias:

- cd PracticasBT
- cd Practica2
- ns movement.tcl >> movement.out



Al presionar *play* en la presente simulación se ejecutará una piconet entre los nodos 0, 1 y 2 durante un tiempo determinado. Los comandos que se utilizarán son los siguientes:

$ns at 1.0 "$node(0) make-bnep-connection $node(2) DH5 DH3 noqos $ifq1"
$ns at 1.1 "$node(0) make-bnep-connection $node(1) DH5 DH3 noqos $ifq"
$ns at 4.0 "$ftp0 start"
$ns at 4.1 "$ftp1 start"

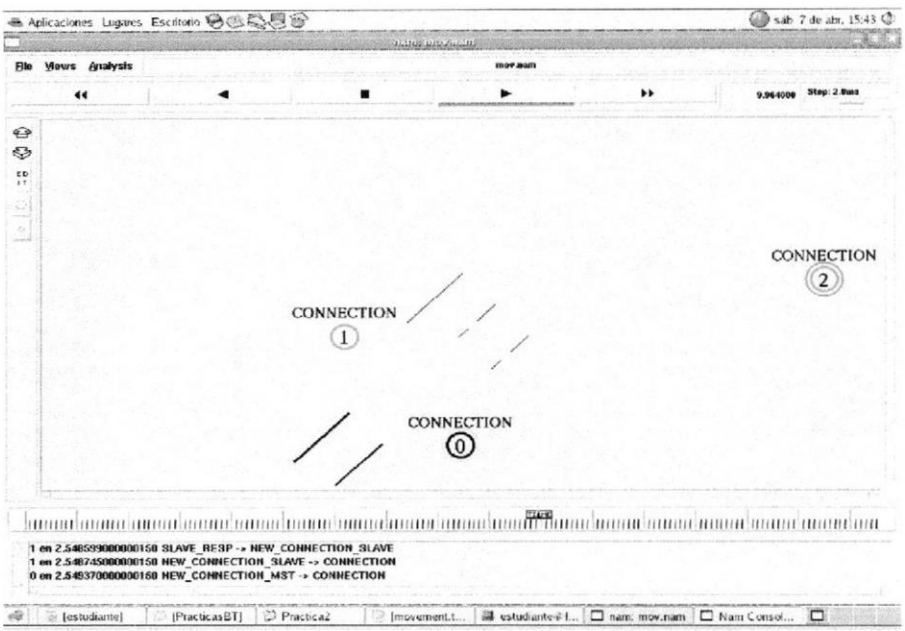


Una vez que la piconet ha sido establecida, los nodos 1 y 2 empezarán a moverse alrededor del nodo 0 o maestro, de tal manera que el nodo 2 termine fuera del rango de cobertura de la piconet y el nodo 1 se mantenga

en el rango de cobertura de la misma. Los comandos que se ejecutarán son los siguientes:

$ns at 4.7 "$node(1) setdest 6 10 1"
$ns at 4.8 "$node(2) setdest 22 8 5.5"

Estos comandos permiten el movimiento de los nodos 1 y 2. Al finalizar esta acción el nodo 2 se encontrará fuera del rango de cobertura y perderá toda transmisión pendiente con el nodo 0 como podemos apreciar en la siguiente figura:



Al finalizar la simulación conteste las siguientes preguntas:

- ¿En que momento el nodo 2 deja de recibir paquetes del nodo 0?
- Indique las posiciones iniciales y finales de los nodos 1 y 2

5.2 Simular dos piconets conectadas entre ellas por un nodo PMP.

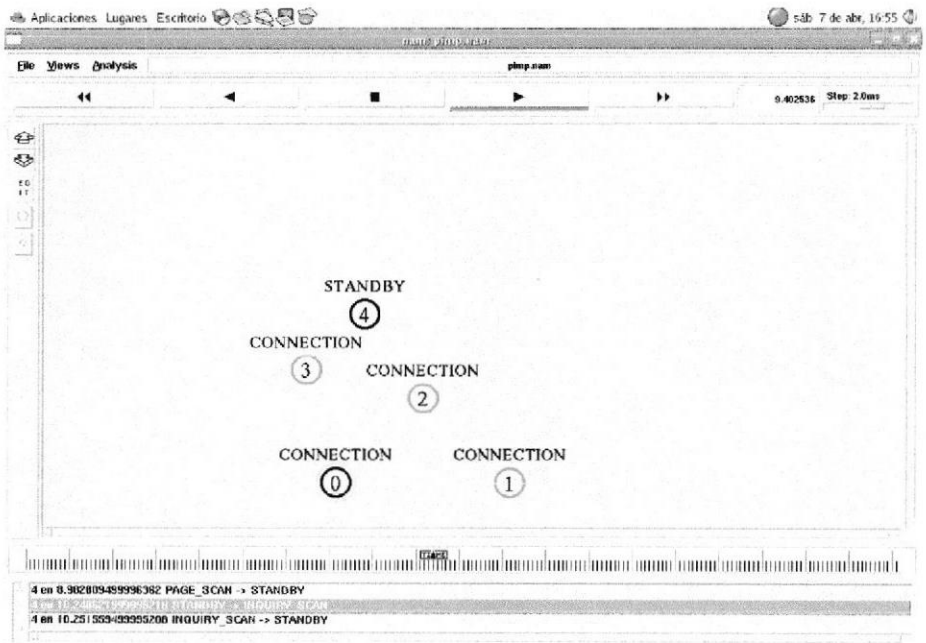
Los nodos PMPs, o también conocidos como nodos participantes en múltiples piconets, son aquellos que establecieron conexión con más de una piconet a la vez. Las razones por las cuáles estos nodos llegaron a participar en varias piconets puede ser: por haber formado parte de un procedimiento de cambio de roles, por haber aceptado aplicaciones de dispositivos que están formando piconets diferentes, o por petición de una topología compleja de red Bluetooth, como una scatternet, haciendo que este dispositivo sirva de nodo puente entre piconets por su situación geográfica.

Una vez que tengamos la pantalla del terminal, ejecutaremos las siguientes sentencias:

- cd PracticasBT
- cd Practica2
- ns pmp.tcl >> pmp.out

Al presionar play en esta simulación tendremos la formación una piconet inicial entre 0, 1, 2 y 3. Los comandos a utilizar son los siguientes:

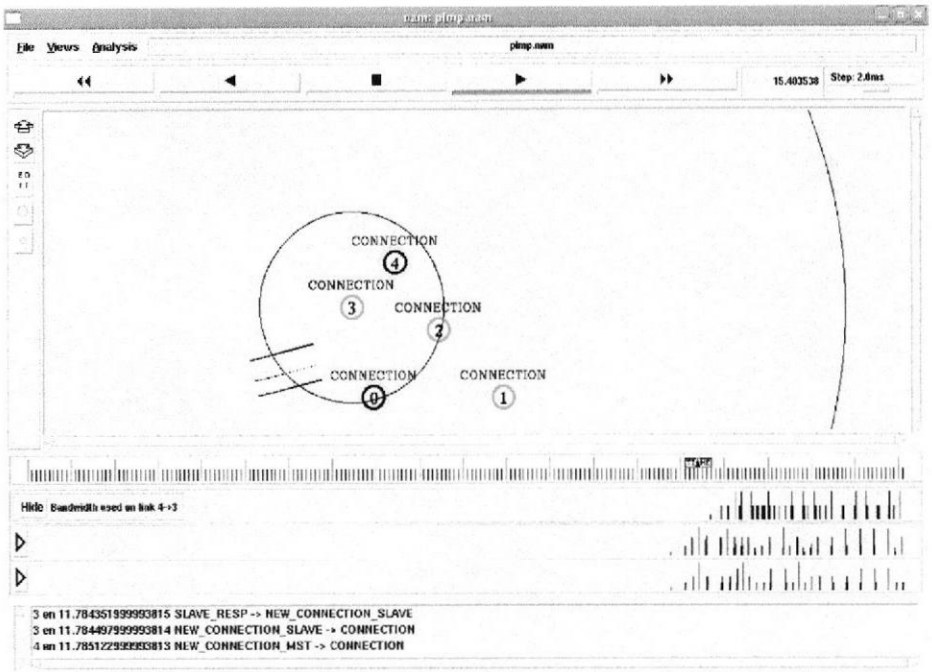
$ns at 1.2 "$node(0) make-bnep-connection $node(1) DH5 DH3 none"
$ns at 3.4 "$node(0) make-bnep-connection $node(2) DH5 DH3 none"
$ns at 5.6 "$node(0) make-bnep-connection $node(3) DH5 DH3 none"
$ns at 14.5 "$ftp0 start"



Para conectar el nodo 4 con la piconet vamos a utilizar el nodo 3 como puente con el siguiente comando:

$ns at 10.5 "$node(4) make-br $node(3) DH5 DH3 none"

Con esta sentencia creamos una nueva piconet entre el nodo 3 y el nodo 4 que a su vez se encuentra conectada con la piconet anterior gracias al nodo 3 que actúa como bridge o puente entre ambas piconets. En la siguiente figura se muestra una transferencia de archivos entre las dos piconets:



Al finalizar la simulación conteste las siguientes preguntas:

- ¿Qué tipo de funciones realiza el nodo PMP?
- ¿En qué momento se establece la piconet entre los nodos 3 y 4?
- ¿Qué rol desempeña el nodo PMP en esta simulación?

5.3 Simular el intercambio de roles entre dos dispositivos conectados en una piconet.

Este procedimiento es utilizado únicamente por dos dispositivos que estén conectados en una piconet, es decir, ambos dispositivos intercambiarán roles, dando como resultado que el dispositivo maestro se convierta en el esclavo y que el esclavo se convierta en maestro. Cuando ocurre este procedimiento se forma nuevamente la piconet, en otras palabras, el esquema de saltos entre canales se redefine según el nuevo dispositivo maestro.

Si existiera el caso en el que uno de los dispositivos que va a realizar el cambio de roles tuviera alguna conexión con otro dispositivo, este enlace no se pierde y el dispositivo quedaría trabajando como un nodo participante en múltiples piconets.

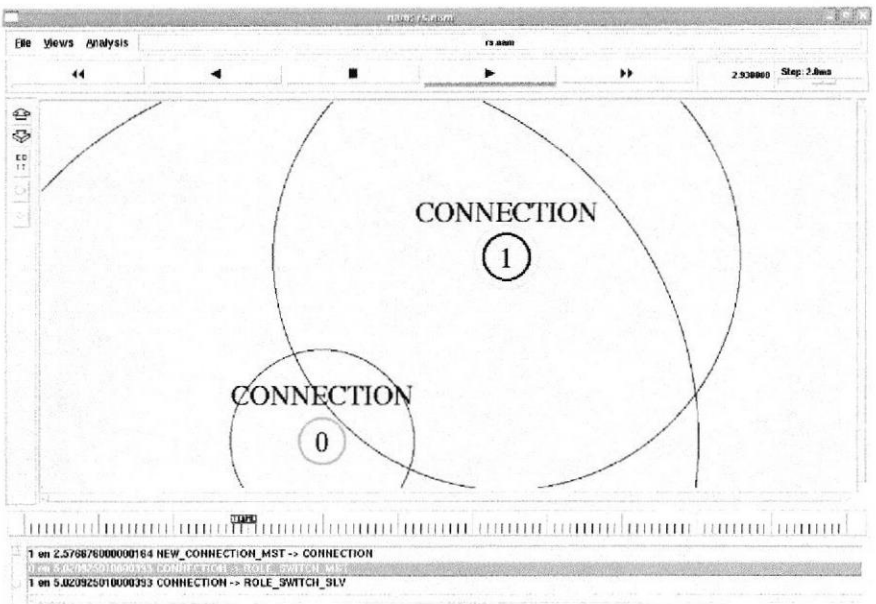
Una vez que tengamos la pantalla del terminal, ejecutaremos las siguientes sentencias:

- cd PracticasBT

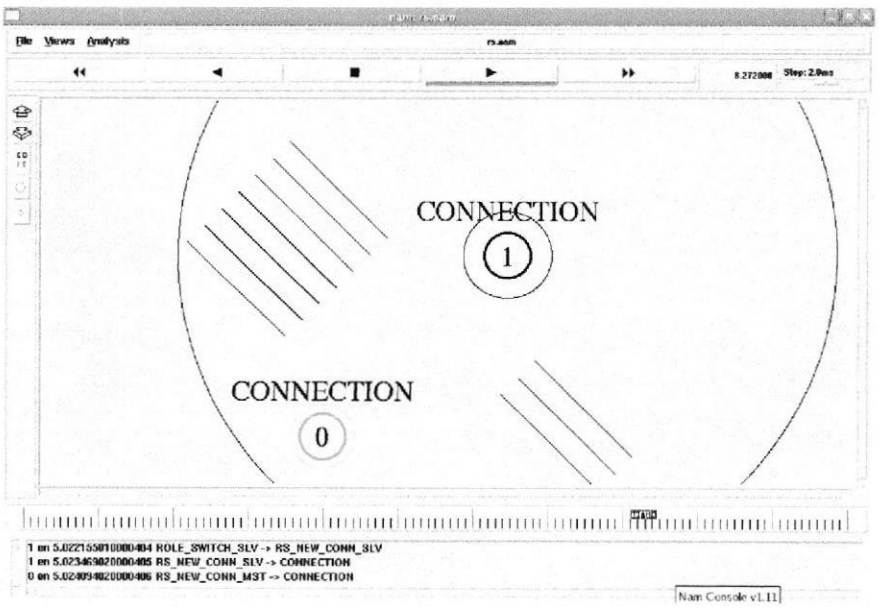
- cd Practica2
- ns rs.tcl >> rs.out

Al ejecutar esta simulación tendremos una piconet entre los nodos 1 y 0, siendo maestro y esclavo respectivamente, utilizando el siguiente comando:

$ns at 0.1 "$node(1) make-bnep-connection $node(0) DH1 DH1"



Una vez establecida la piconet procedemos al intercambio de roles entre el nodo 0 y el nodo 1, así también se establecerá una transferencia TCP entre ambos nodos teniendo ahora como nuevo maestro al nodo 0 y como nuevo esclavo al nodo 1 como lo observamos en la siguiente figura:



Al finalizar la simulación conteste las siguientes preguntas:

- Verificar los canales de frecuencia que se utilizan en la piconet antes de que exista el intercambio de roles.
- Verificar todos los estados y paquetes intercambiados en el momento del intercambio de roles.
- Verificar los canales de frecuencia que usa la nueva piconet. ¿Existe alguna diferencia?
- Verifique el CLK que utiliza la nueva piconet.

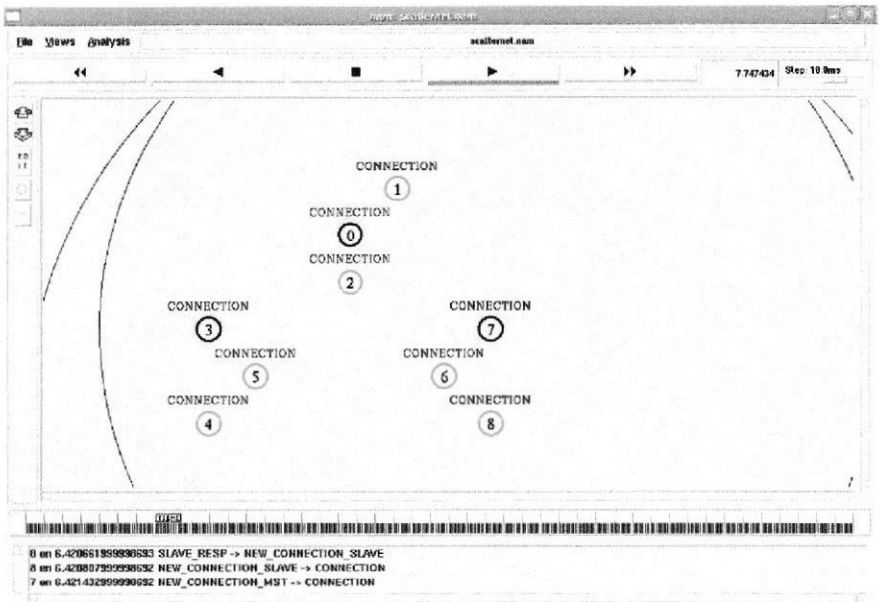
5.4 Simular una red Bluetooth avanzada (Scatternet).

Las scatternets básicamente resultan de la interconexión de 2 o más piconets. Es una topología de red bastante compleja, ya que se pueden solapar ciertos canales e incurrir a niveles de interferencias considerables entre las piconets que se interconecten, así como problemas en la transmisión de datos, ancho de banda y sincronización de los dispositivos que participan en la scatternet. Además, los nodos que enlazan estas piconets, conocidos como nodos puentes, deben participar en ambas piconets según el rol que tengan en cada una.
Una vez que tengamos la pantalla del terminal, ejecutaremos las siguientes sentencias:

- cd PracticasBT
- cd Practica2
- ns scatternet.tcl >> scatternet.out

Al ejecutar esta simulación estableceremos tres piconets entre los nodos 0, 1 y 2 la primera, 3, 4 y 5 la segunda y 6, 7 y 8 la tercera con los siguientes comandos:

$ns at 1.2 "$node(0) make-bnep-connection $node(1) DH5 DH3 none"
$ns at 1.4 "$node(0) make-bnep-connection $node(2) DH5 DH3 none"
$ns at 2.2 "$node(3) make-bnep-connection $node(4) DH5 DH3 none"
$ns at 2.4 "$node(3) make-bnep-connection $node(5) DH5 DH3 none"
$ns at 3.2 "$node(7) make-bnep-connection $node(6) DH5 DH3 none"
$ns at 3.4 "$node(7) make-bnep-connection $node(8) DH5 DH3 none"



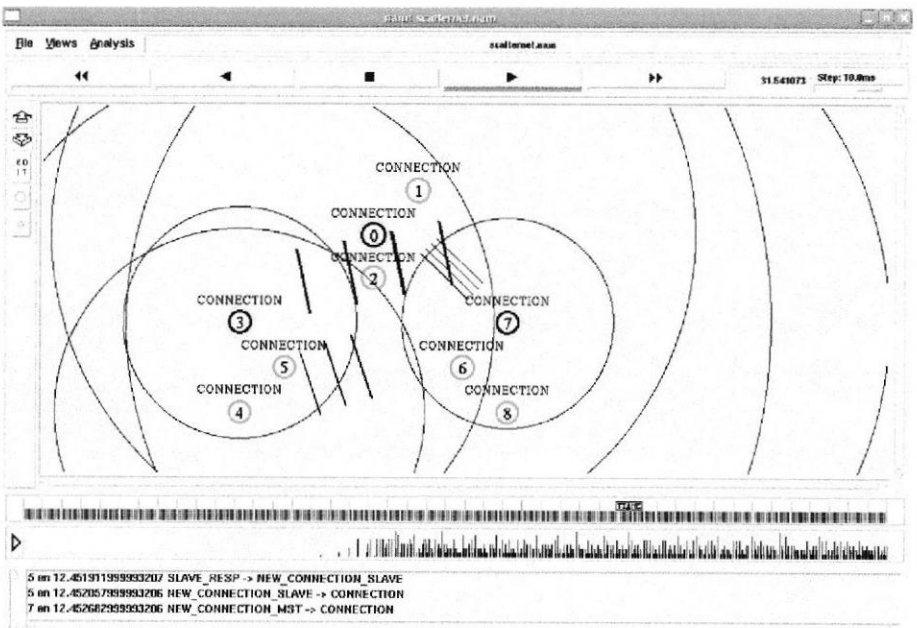
Luego procederemos a interconectar las tres piconets mediante dos nodos puente con el siguiente comando:

$ns at 10.0 "$node(3) make-br $node(2) DH5 DH3 none"
$ns at 11.0 "$node(7) make-br $node(5) DH5 DH3 none"

Finalizaremos con una transferencia de datos entre los nodos 0 y 7 para verificar que las piconets se encuentren interconectadas por medio del siguiente comando:

$ns at 15.0 "$ftp0 start"

En la siguiente figura vemos la transferencia de datos entre ambos nodos:



BIBLIOGRAFIA

The VINT Project, The ns Manual, UC Berkeley, LBL, USC/ISI, and Xerox PARC,.Kevin Fall (kfall@ee.lbl.gov), Kannan Varadhan kannan@catarina.usc.edu), Editores; Enero 8, 2003.

BluetoothSIG, Specifications Documents, http://www.bluetooth.com/Bluetooth/Learn/Technology/Specifications/ revisada el 4 de Enero 2006.

D. Agrawal, Q. Wang, UCBT Bluetooth Extension for NS2 at the University of Cincinnati, http://www.ececs.uc.edu/~cdmc/ucbt/ucbt.html revisada el 15 de Enero de 2006.

APÉNDICE C.

Guías resueltas de las prácticas del laboratorio para los tutores.

C.1 Guía de Práctica No. 1 "Simulación de procedimientos básicos para la conexión de dispositivos Bluetooth basados en NS2/UCBT/NAM"

Guía de Práctica No.1

Simulación de procedimientos básicos para la conexión de dispositivos Bluetooth basados en NS2/UCBT/NAM

1. OBJETIVOS

- Orientar a los estudiantes al manejo de las simulaciones de redes Bluetooth utilizando el simulador de redes NS2.
- Conocer y visualizar los procedimientos que están incluidos en la formación de redes Bluetooth.
- Crear una topología de red básica Bluetooth (Piconet).
- Verificar el número máximo de enlaces activos dentro de una piconet.

2. INTRODUCCION

Dentro de esta práctica se aprenderán varios procedimientos básicos para establecer una comunicación vía Bluetooth, dentro de los cuales se mencionan: el Inquiry o Descubrimiento, el Paging o Emparejamiento, y el intercambio de paquetes necesarios para poder establecer una comunicación entre dispositivos.

UCBT es una librería compatible con el simulador de redes NS2. El objetivo de esta librería es utilizar los nodos creados en NS2 y adaptarlos para que funcionen dentro de la especificación Bluetooth, en lo que se refiere a conexión, transferencia de archivos, rango de cobertura, movilidad, entre otros.

3. MARCO TEORICO

Bluetooth es una norma que define un standard global de comunicación inalámbrica, que posibilita la transmisión de voz y datos entre diferentes equipos mediante un enlace por radiofrecuencia. Los dispositivos Bluetooth operan en una frecuencia de radio que se encuentra entre los 2.4 y 2.48GHz, con la posibilidad de realizar hasta 1.600 saltos por segundo entre las 79 frecuencias soportadas en intervalos de 1MHz. Cada unidad incluye una radio, un controlador de enlaces de banda base y el software para la administración de los enlaces y flujo de datos.

Los usuarios tienen la opción de dos potencias de señal: un nivel de baja potencia para distancias de hasta 10 metros, y un nivel de alta potencia de hasta 100 metros de distancia para los puntos de acceso. Los dispositivos Bluetooth pueden conectarse simultáneamente hasta siete aparatos más, sin incluir el dispositivo maestro. La velocidad máxima de transferencia de datos es de aproximadamente 720 Kbps por canal.

4. MATERIALES Y EQUIPOS

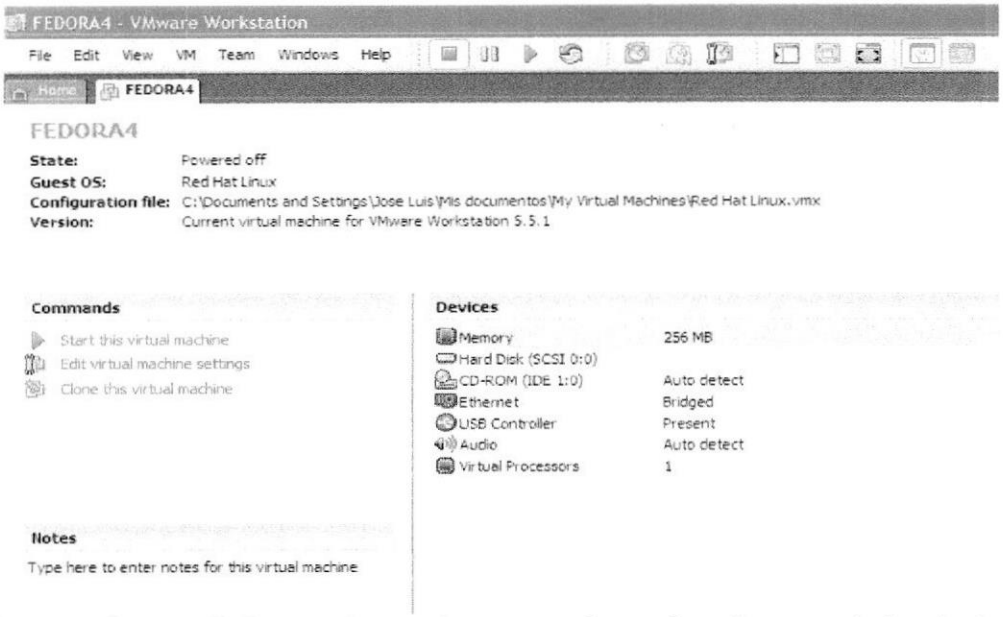
- PC con Fedora Core 4 y Network Simulator 2: Network Animator, librería UCBT modificada.

5. PROCEDIMIENTOS

Abriremos el emulador de sistemas operativos "VMware Workstation", para esto haga doble click en el icono **VMware Workstation 5.5.1** en el escritorio o vaya a la siguiente dirección:

Inicio->Programas->VMware->VMware Workstation 5.5.1

aparecerá la siguiente ventana:



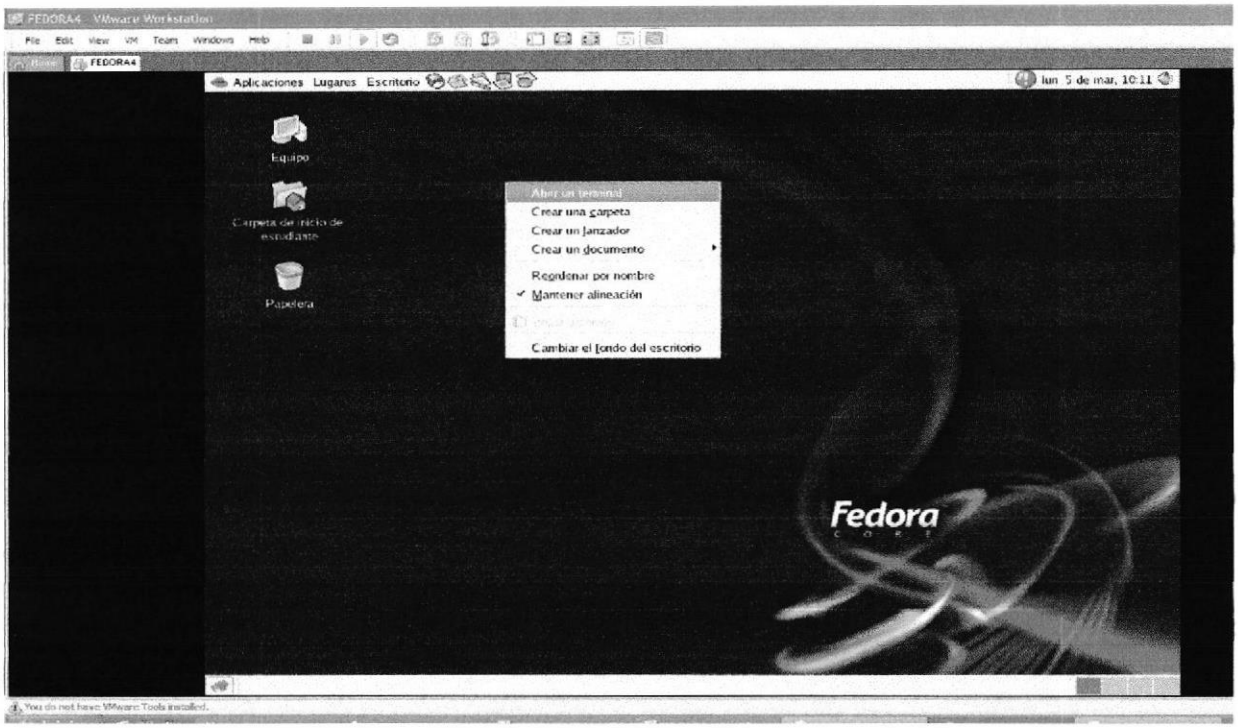
desde aquí haga click en "*Start this virtual machine*" para abrir el sistema operativo "Fedora Core 4".

En la pantalla de inicio de sesión de Fedora Core 4 proceda a digitar en el campo usuario lo siguiente: Estudiante, como se muestra a continuación:

Luego, en el campo contraseña, proceda a digitar lo siguiente: Estudiante, y al finalizar presione "*Enter*".

5.1 Simular un procedimiento de Inquiry y observar los cambios de estados de los nodos.

Dentro del escritorio abrimos una ventana de terminal, esto lo hacemos dando click derecho sobre el escritorio

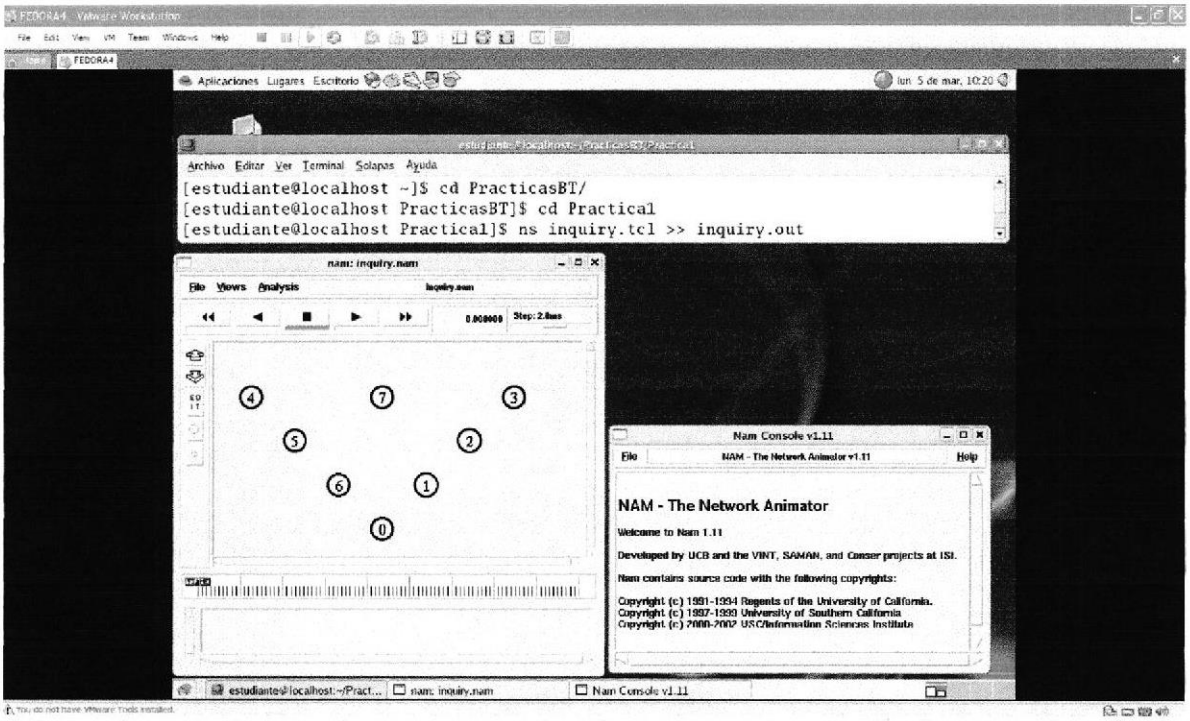


Una vez que tengamos la pantalla del terminal, ejecutaremos las siguientes sentencias:

cd PracticasBT

cd Practica1

ns inquiry.tcl >> inquiry.out



Al presionar *play* en la presente simulación se ejecutarán los estados de descubrimiento de dispositivos en los nodos 0 y 7, durante un tiempo determinado. El comando que se utilizará es el siguiente:

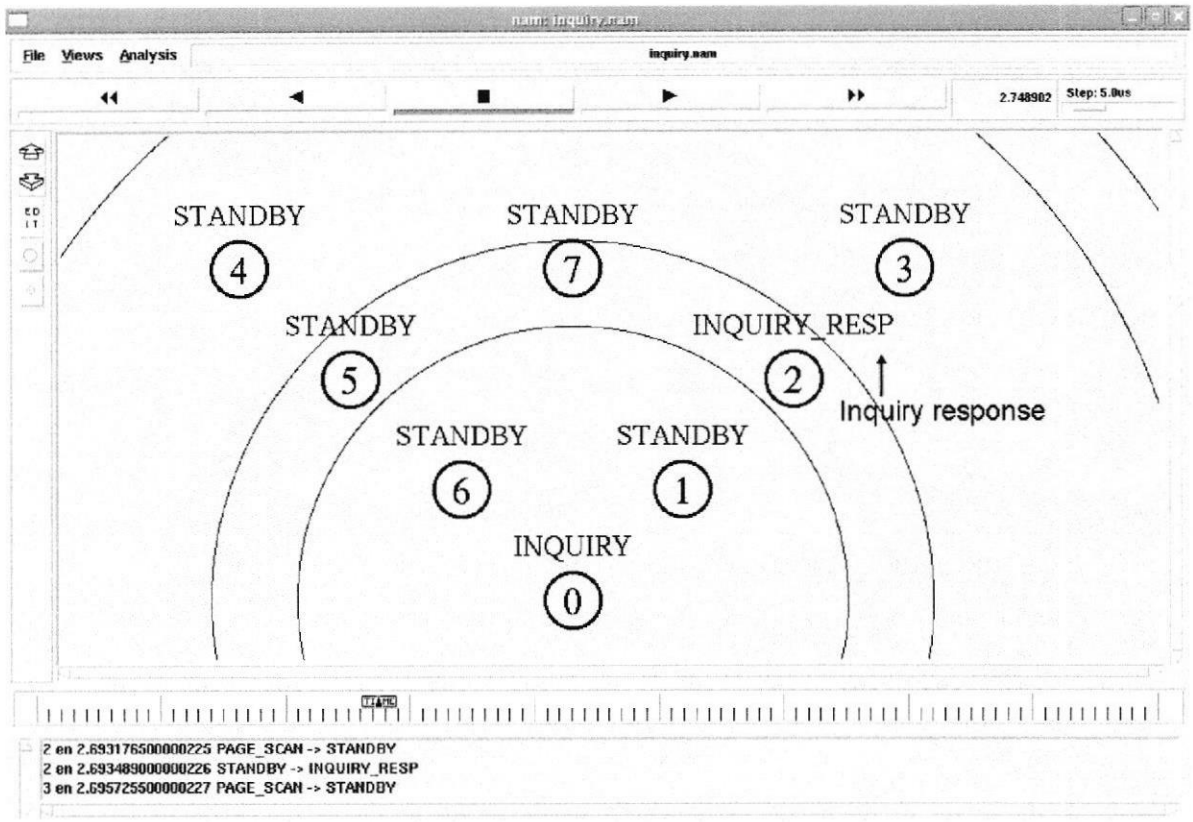
$ns at 0.1 "$node(0) inquiry 5 5"

Lo que implica que el nodo 0 permanecerá en un estado de inquiry durante 5x1.28seg. a menos que reciba 5 inquiry responses de dispositivos diferentes. Para el nodo 7 se ejecutará la siguiente sentencia:

$ns at 6.0 "$node(7) inquiry 2 5"

Se recomienda modificar los parámetros de respuestas de inquiry y el tiempo que toma el realizar el procedimiento para generar diferentes resultados en las simulaciones. Además se pueden escoger diferentes nodos para que realicen las peticiones de inquiry.

En la siguiente figura se puede apreciar el momento en el cual el nodo 2 envía un inquiry response hacia el nodo 0.



Al finalizar esta simulación conteste las siguientes preguntas:

¿Cuáles son los 5 nodos que envían *"inquiry responses"* al nodo 0 y en que tiempo ocurre cada respuesta?

Los nodos que envían "inquiry responses al nodo 0 son: el nodo 1 en 0.2536s, el nodo 2 en 0.3861, el nodo 3 en 2.8670, el nodo 4 en 3.3073 y el nodo 5 en 3.8845

¿Cuánto tiempo se toma el nodo 0 durante el procedimiento de inquiry?

El nodo 0 empieza el procedimiento de inquiry en el tiempo 0.1001s y termina en el tiempo 3.9841s

¿Cuáles son los nodos que envían *"inquiry responses"* al nodo 7?

Los nodos que envían "inquiry responses" al nodo 7 son: el nodo 2 en el tiempo 8.3043s y el nodo 4 en el tiempo 8.3068s

¿Cuanto tiempo se toma el nodo 7 durante el proceso de Inquiry?

El nodo 7 empieza el procedimiento de inquiry en el tiempo 6.0003s y termina en el tiempo 8.5606s, es decir el nodo 7 se toma 2*1.28=2.56 segundos.

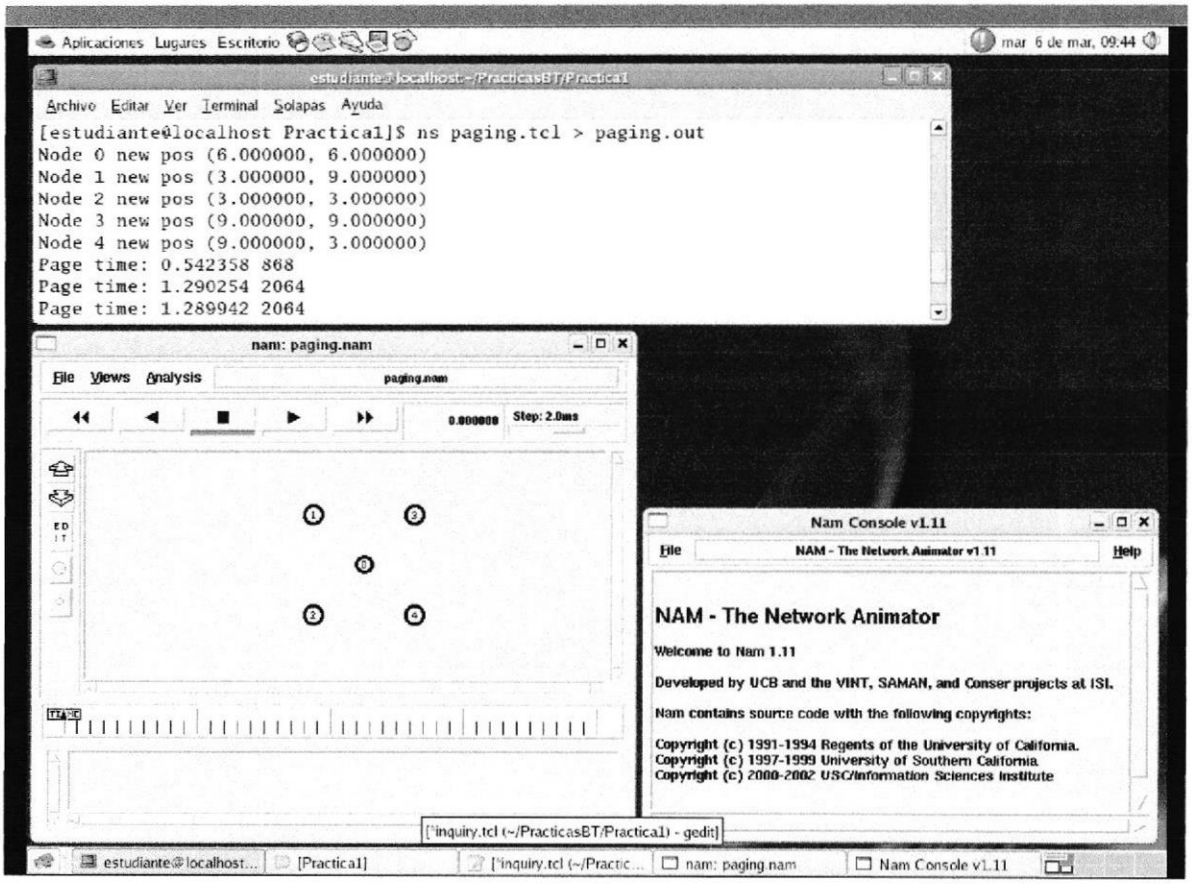
*Cabe recalcar al estudiante que para el nodo 7 se puede utilizar la fórmula T_Inq=n*1.28s debido a que no recibe al menos cinco respuestas durante este tiempo.*

5.2 Simular un procedimiento de Paging y observar los roles con los que quedan los nodos

Una vez que conocemos los nodos disponibles para conectarse, se realiza el procedimiento de conexión o Paging, donde los nodos intercambiarán información con respecto a su posición y su frecuencia de reloj CLK para proceder a sincronizarse a un esquema dominante. El nodo que se impone en este esquema se denominará "maestro" y los que lo seguirán se llamarán "esclavos".

En la ventana del terminal proceda a ejecutar la siguiente sentencia:

- ns paging.tcl >> paging.out



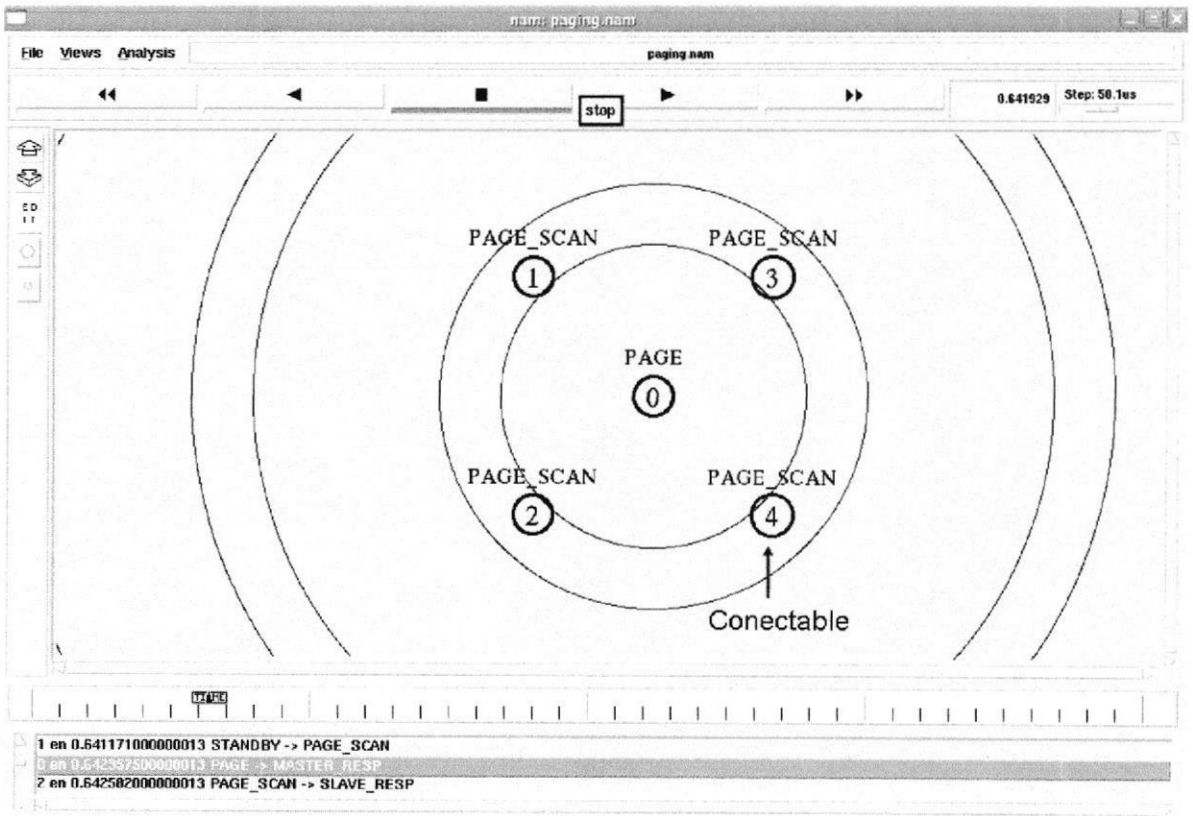
Al presionar el botón *play* en la siguiente simulación, podemos observar que se realizara el procedimiento de conexión de dispositivos, paging, dando como resultado un dispositivo maestro y varios dispositivos esclavos. Esto se lo consigue con el siguiente comando:

$ns at 0.1 "$node(0) make-bnep-connection $node(1)"

Este comando permite establecer una conexión hasta la capa BNEP dentro de la pila de protocolos Bluetooth entre el nodo 0 y el nodo 1. Ambos dispositivos entrarán en el procedimiento de paging y de page scan respectivamente, realizando intercambios de paquetes necesarios para la conexión de ambos dispositivos. La sintaxis de este comando especifica que el nodo cero logrará el rol de maestro y el nodo 1 logrará el rol de esclavo respectivamente.

Se recomienda modificar el orden de los nodos en el comando "make-bnep-connection" para obtener diferentes respuestas con lo que respecta a que nodos lograrán el rol de maestro y esclavo. Así como diferenciar el tipo de paquetes que intercambiarán para este nuevo caso.

En la figura se puede apreciar a los dispositivos en el momento en el que se envían respuestas para los dispositivos



Al finalizar esta simulación conteste las siguientes preguntas:

- **¿Qué tipo de paquetes envía y recibe el nodo 0?**
  El nodo 0 envía paquetes de tipo ID, POLL y FH mientras que recibe paquetes de tipo ID y NULL.
- **¿Qué tipo de paquetes envían y reciben los otros nodos?**

Los nodos 1, 2, 3 y 4 envían paquetes de tipo ID y NULL mientras que reciben paquetes de tipo ID, FH y POLL.

- **¿Qué nodos logran el rol de maestro?**
  Solamente el nodo 0 logra el rol de maestro.
- **¿Qué nodos logran el rol de esclavo?**
  Los nodos 1, 2, 3 y 4 logran el rol de esclavo.
- **¿Por cuáles estados deben de pasar los nodos hasta llegar al estado de conexión? Haga una tabla con cada nodo.**

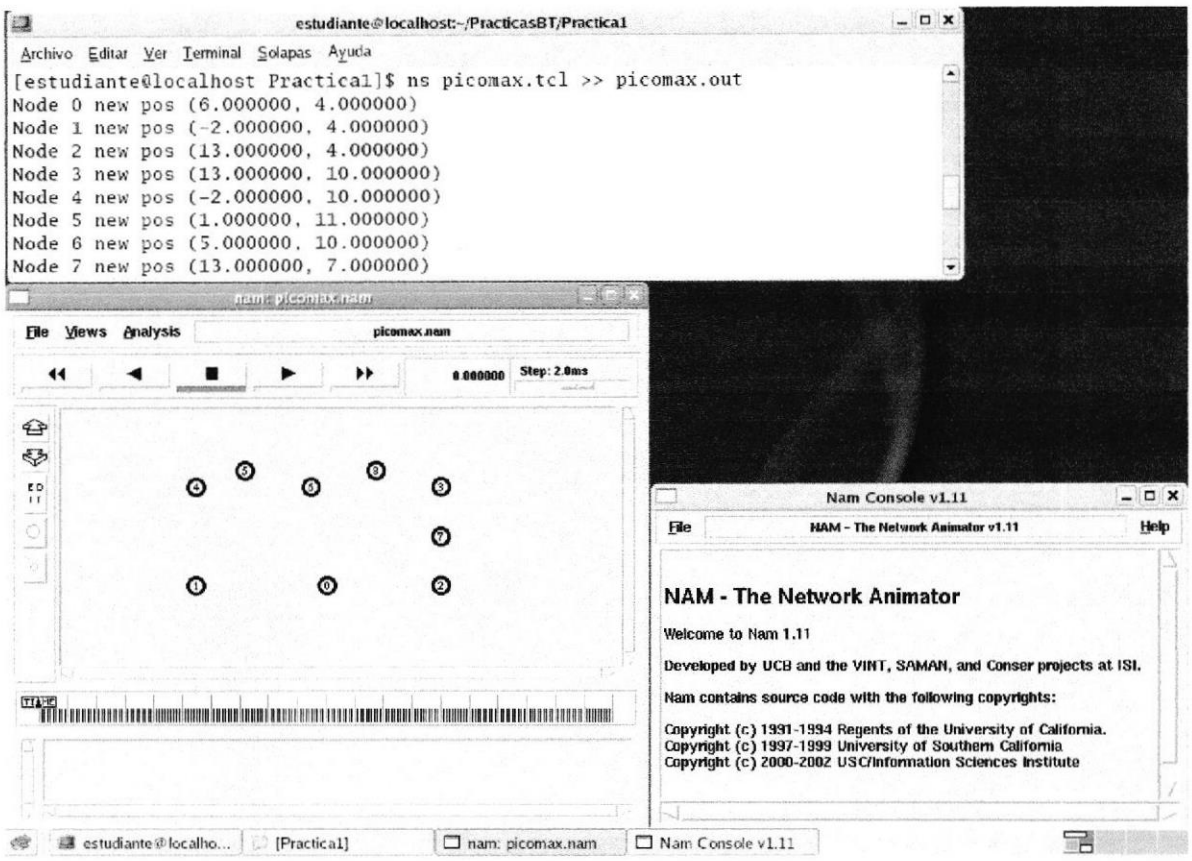
Nodo 0	**Nodo 1**	**Nodo 2**	**Nodo 3**	**Nodo 4**
Inquiry scan	*Inquiry Scan*	*Inquiry Scan*	*Inquiry Scan*	*Inquiry Scan*
Standby	*Standby*	*Standby*	*Standby*	*Standby*
Page	*Page Scan*	*Page Scan*	*Page Scan*	*Page Scan*
Master response	*Slave Response*	*Slave Response*	*Slave Response*	*Slave Response*
New Connetion master	*New connection Slave*	*New connection Slave*	*New connection Slave*	*New connection Slave*
Connection	*Connection*	*Connection*	*Connection*	*Connection*

5.3 Simular una piconet con la cantidad máxima de nodos conectados y con una transferencia de archivos bajo una aplicación TCP.

Una vez concluido el procedimiento de paging, los dispositivos tomaron roles específicos dentro de una conexión común llamada piconet. Una piconet consta de un único dispositivo maestro al cual se conectan varios dispositivos esclavos. El número máximo de dispositivo esclavos "activos" dentro de una piconet es de 7. Una vez conectados, los nodos pueden intercambiar archivos según aplicaciones que se demanden entre dispositivos. Estas aplicaciones pueden ser basadas en el protocolo TCP o UDP según la aplicación.

En la ventana del terminal proceda a ejecutar la siguiente sentencia:

- ns picomax.tcl >> picomax.out



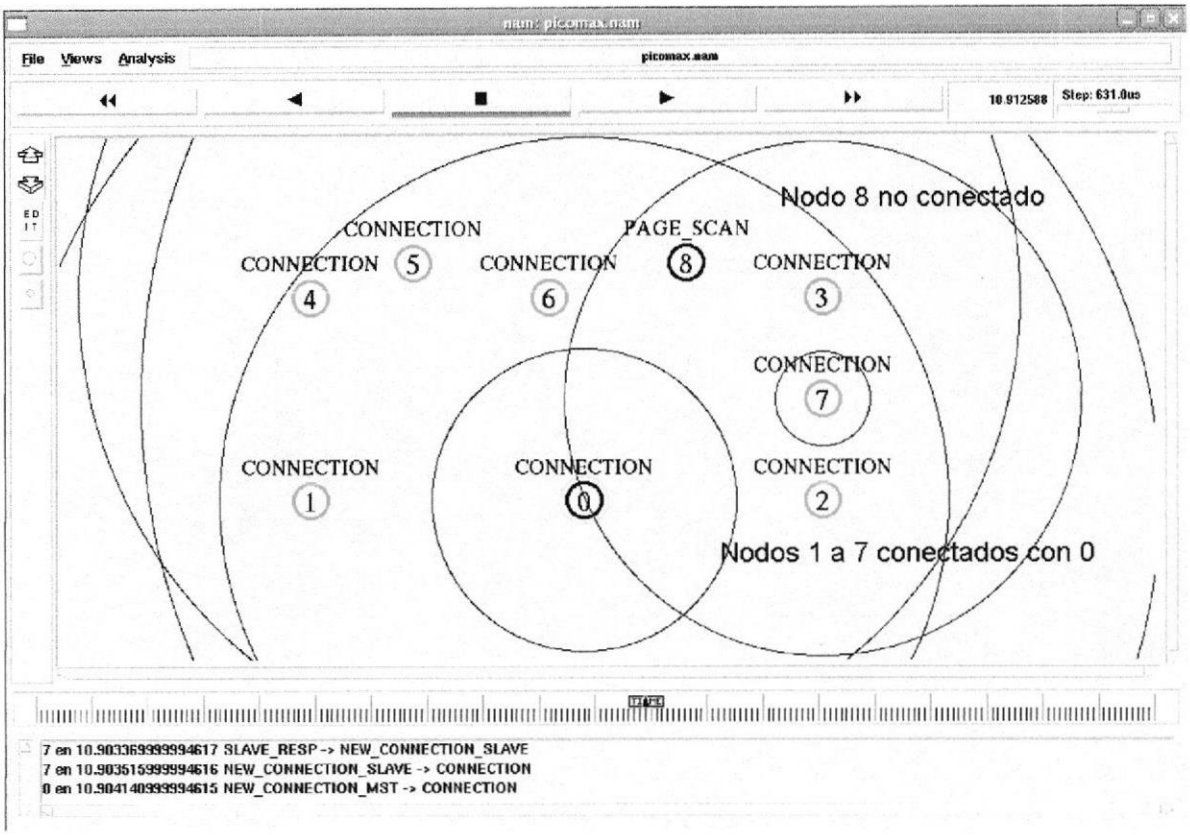
Al presionar el botón *play* en la presente simulación, podemos observar la formación de una piconet con el número máximo de nodos esclavos activos (7). Además habrá una transferencia de tipo TCP entre los nodos 1 y 2 que se la consigue con las siguientes líneas de código:

$ns at 0.1 "$node(0) make-bnep-connection $node(1) DH3 DH3 noqos $ifq"
$ns at 0.2 "$node(0) make-bnep-connection $node(2)"
$ns at 12 "$ftp0 start"

Estas líneas de código permiten establecer una aplicación de tipo FTP entre los nodos 1 y 2 mediante una transferencia de archivos de tipo TCP. Antes de empezar el traspaso de datos, se debe realizar una conexión hasta la capa BNEP entre el nodo maestro 0 y los nodos esclavos 1 y 2, respectivamente.

Se recomienda comprobar la conectividad del nodo ocho agregándolo a la piconet en lugar de un nodo que ya forma parte de la red. Para esto, modifique el comando "make-bnep-connection" con un nodo que no participe en la transferencia de datos de la piconet.

En la siguiente figura podemos apreciar la formación de una piconet:



Al finalizar esta simulación conteste las siguientes preguntas:

- **Explique la función del nodo con marca amarilla y de los nodos con marca verde.**
  El nodo con marca amarilla es considerado el maestro de la piconet y su función consiste en sincronizar y establecer conexiones con los nodos que lo demanden. Los nodos con marca verde son considerados los esclavos de la piconet y su función consiste en intercambiar con el maestro información de su estado tales como su posición y conectividad.
- **Explique que sucede con el nodo 8.**
  El nodo 8 no puede conectarse a la piconet debido a que ya existen 7 nodos esclavos activos conectados.
- **En la transferencia de paquetes entre los nodos 1 y 2, explique por qué el tráfico pasa por el nodo 0.**
  Debido a que los nodos 1 y 2 son esclavos, el tráfico de paquetes debe pasar por el nodo maestro que los conecta, en este caso, el nodo 0.

BIBLIOGRAFIA

1. The VINT Project, The ns Manual, UC Berkeley, LBL, USC/ISI, and Xerox PARC,.Kevin Fall (kfall@ee.lbl.gov), Kannan Varadhan kannan@catarina.usc.edu), Editores; Enero 8, 2003.

2. BluetoothSIG, Specifications Documents, http://www.bluetooth.com/Bluetooth/Learn/Technology/Specifications/ revisada el 4 de Enero 2006.

3. D. Agrawal, Q. Wang, UCBT Bluetooth Extension for NS2 at the University of Cincinnati, http://www.ececs.uc.edu/~cdmc/ucbt/ucbt.html revisada el 15 de Enero de 2006.

C.2 Guía de Práctica No. 2 "Transferencia de paquetes en diferentes topologías de redes Bluetooth basados en NS2/UCBT/NAM"

Práctica No.2

Transferencia de paquetes en diferentes topologías de redes Bluetooth basados en NS2/UCBT/NAM

1. OBJETIVOS

- Examinar escenarios móviles de redes Bluetooth y verificar sus rangos de cobertura.
- Conocer el funcionamiento de los nodos PMPs y su importancia en las redes Bluetooth.
- Mostrar el intercambio de roles de dos dispositivos conectados a una piconet.
- Formar una topología de red Bluetooth avanzada (Scatternet).

2. INTRODUCCIÓN

En esta práctica se aprenderán escenarios avanzados de redes Bluetooth, donde tenemos topologías con gran número de nodos y varias aplicaciones simultáneas, así como el intercambio de archivos entre nodos bajo aplicaciones FTP y CBR y procedimientos de intercambio de roles entre dispositivos conectados.

En la librería UCBT se ha especificado como distancia máxima entre dispositivos Bluetooth cerca de 10 metros a la redonda, es decir la cobertura máxima de cada nodo será esa distancia. El objetivo es utilizar los nodos con las potencias mínimas, teniendo en consideración que por el hecho de ser dispositivos móviles, estos requieren un sistema de ahorro de energía.

3. MARCO TEORICO

Un dispositivo Bluetooth puede atender más de una conexión a la vez, y esto lo logra porque cuando se establece una piconet se usan ciertos canales en un esquema de salto de frecuencia, por lo que cada piconet tendrá un esquema propio. Estos dispositivos que participan en más de una piconet a la vez se los conoce como nodos PMP (Participante en Múltiples Piconets), estos nodos suelen desempeñar tareas importantes cuando de interconectar piconets se trata, es decir, forman un enlace o puente para la interconexión entre piconets. A lo que se conoce con el nombre de scatternet.

Muchas veces dentro del establecimiento de una piconet o incluso de una scatternet, los dispositivos requieren hacer un cambio de roles para poder mejorar su topología y establecer una conexión con la menor cantidad de problemas. A este procedimiento se lo conoce con el nombre de Role-Switch.

Un nodo puente puede tener roles de maestro o de esclavo, según como se realice la interconexión. Debido a esto y otros factores, las tasas de datos en este tipo de topología pueden disminuir según sea el nivel de interferencia o la cantidad de dispositivos y aplicaciones simultáneas que se atiendan. Lo bueno de este tipo de topología es que de alguna manera puede ampliar la cobertura entre dispositivos que se encuentren muy lejanos a los 10 metros.

4. MATERIALES Y EQUIPOS

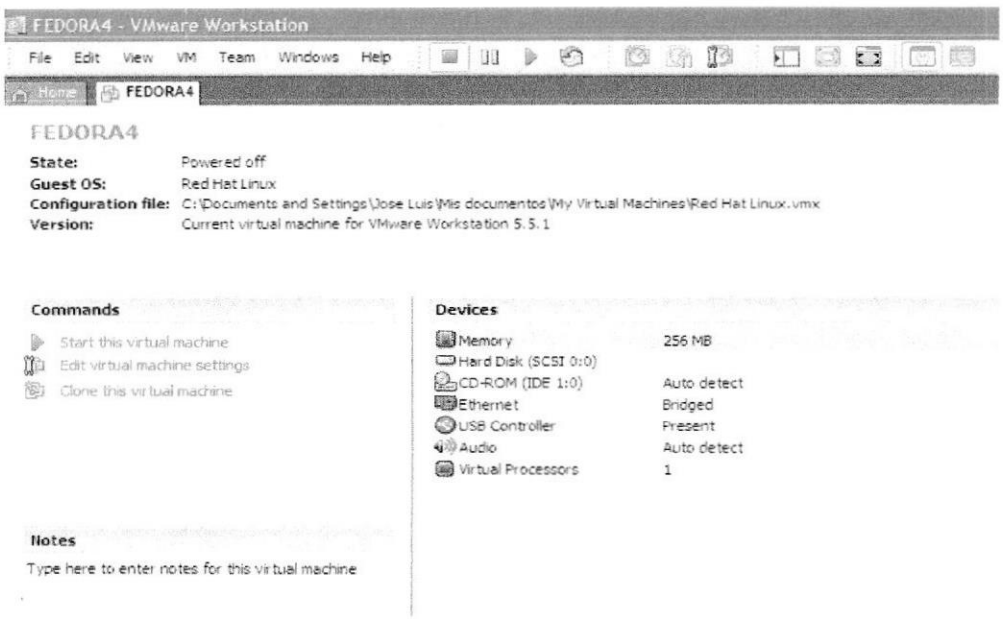
- PC con Fedora Core 4 y Network Simulator 2: Network Animator, librería UCBT modificada.

5. PROCEDIMIENTOS

Abriremos el emulador de sistemas operativos "VMware Workstation", para esto haga doble click en el icono **VMware Workstation 5.5.1** en el escritorio o vaya a la siguiente dirección:

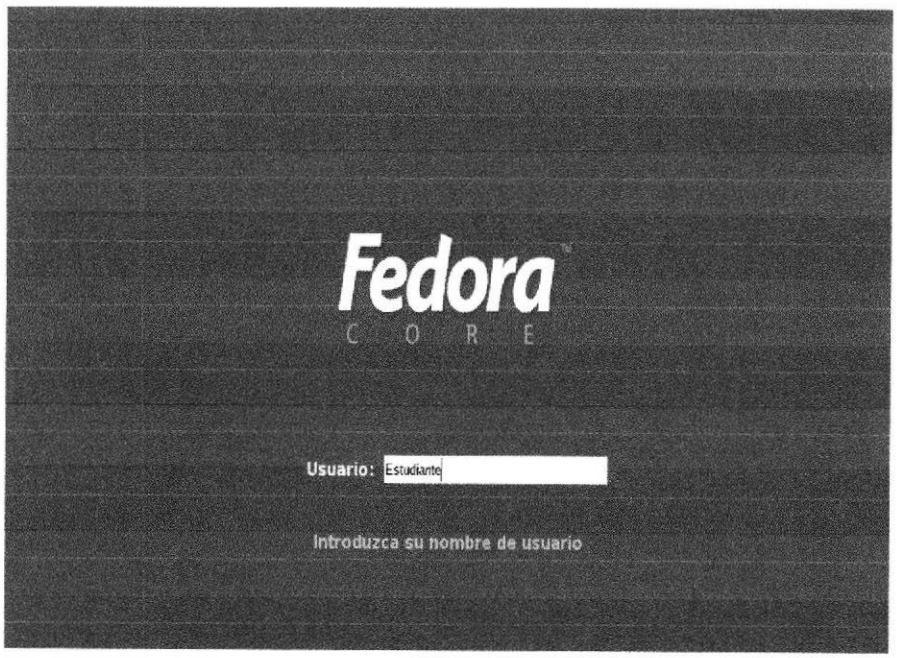
Inicio->Programas->VMware->VMware Workstation 5.5.1

aparecerá la siguiente ventana:

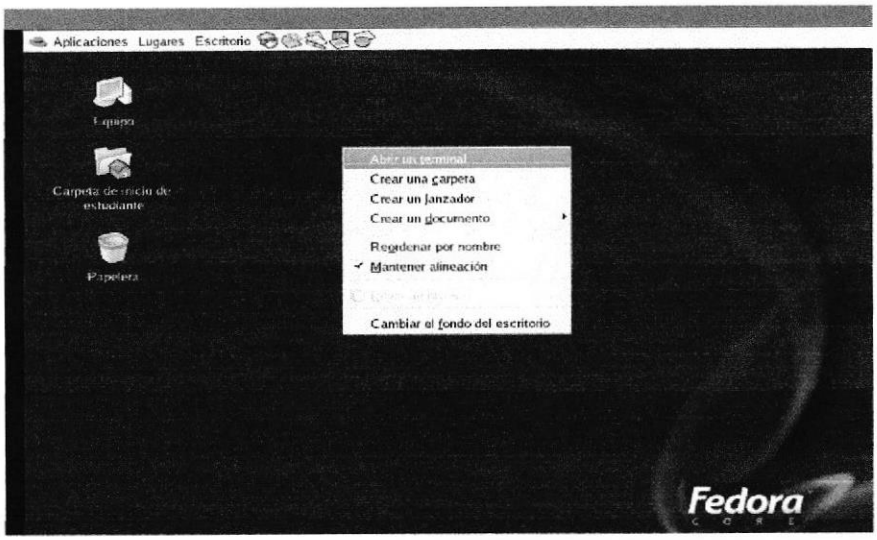


desde aquí haga click en "*Start this virtual machine*" para abrir el sistema operativo "Fedora Core 4".

En la pantalla de inicio de sesión de Fedora Core 4 proceda a digitar en el campo usuario lo siguiente: Estudiante, como se muestra a continuación:



Luego, en el campo contraseña, proceda a digitar lo siguiente: Estudiante, y al finalizar presione "*Enter*". Dentro del escritorio abrimos una ventana de terminal dando click derecho sobre el escritorio.

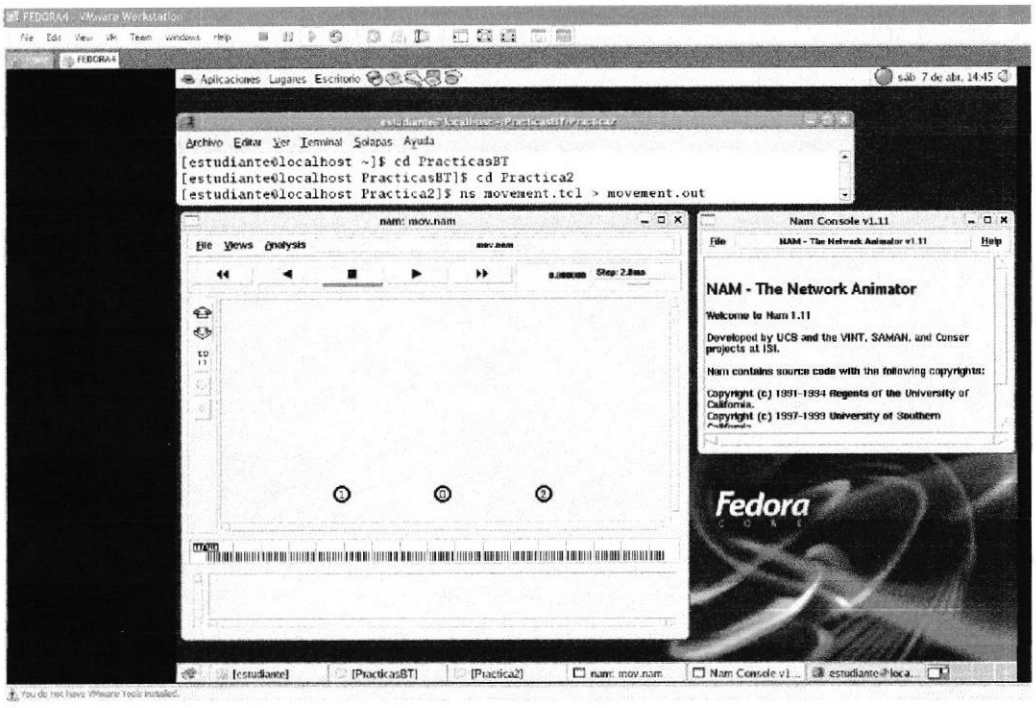


5.1 Simular una piconet con nodos móviles y analizar los rangos de cobertura.

En todo momento, los dispositivos conectados en una piconet deben estar dentro de un espacio físico en el cual puedan comunicarse, es decir, logren un intercambio efectivo de paquetes de información y control. A este espacio físico se lo denomina área de cobertura. Por definición, todos los nodos tienen un área de cobertura de 10 metros para potencias bajas, debido a esto, cuando los dispositivos han establecido una piconet significa que entre el dispositivo maestro y los dispositivos esclavos existen como máximo una distancia de 10 metros.

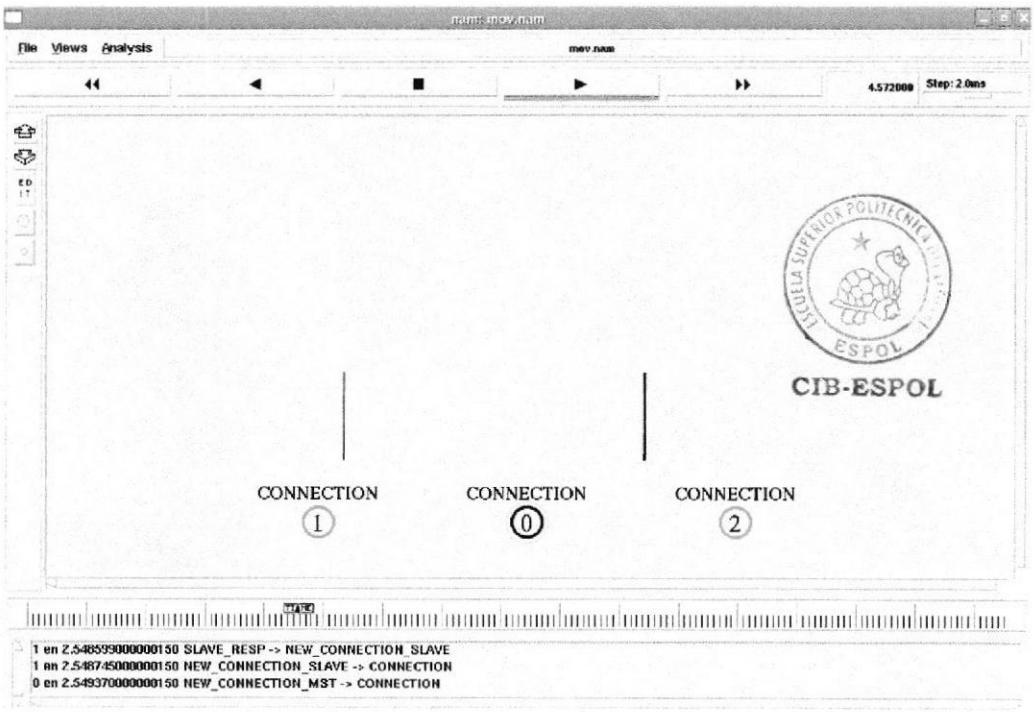
Una vez que tengamos la pantalla del terminal, ejecutaremos las siguientes sentencias:

- cd PracticasBT
- cd Practica2
- ns movement.tcl >> movement.out



Al presionar *play* en la presente simulación se ejecutará una piconet entre los nodos 0, 1 y 2 durante un tiempo determinado. Los comandos que se utilizarán son los siguientes:

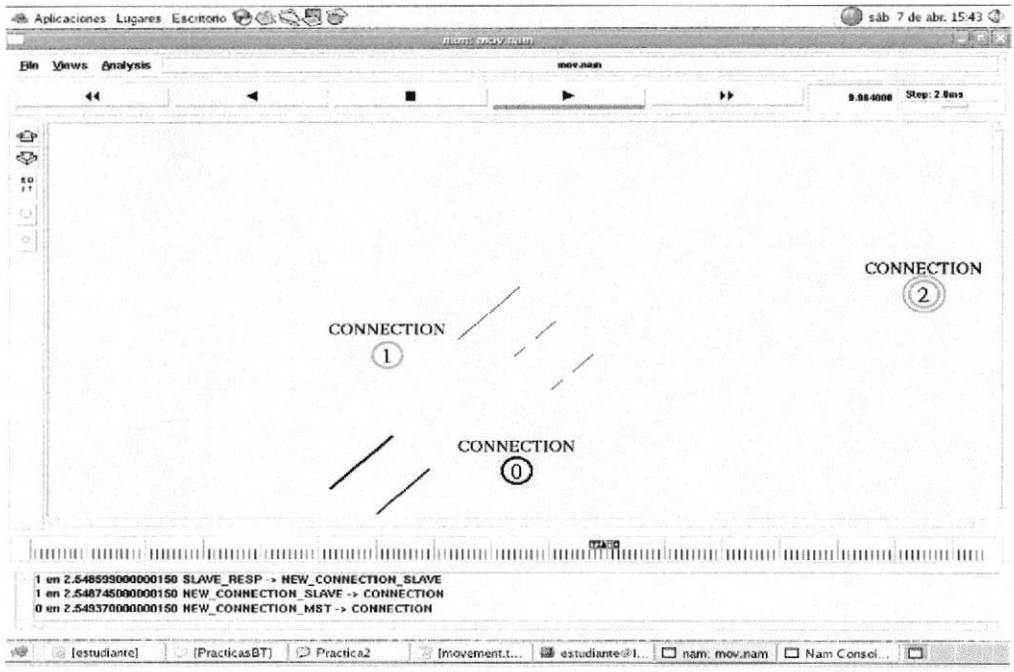
$ns at 1.0 "$node(0) make-bnep-connection $node(2) DH5 DH3 noqos $ifq1"
$ns at 1.1 "$node(0) make-bnep-connection $node(1) DH5 DH3 noqos $ifq"
$ns at 4.0 "$ftp0 start"
$ns at 4.1 "$ftp1 start"



Una vez que la piconet ha sido establecida, los nodos 1 y 2 empezarán a moverse alrededor del nodo 0 o maestro, de tal manera que el nodo 2 termine fuera del rango de cobertura de la piconet y el nodo 1 se mantenga en el rango de cobertura de la misma. Los comandos que se ejecutarán son los siguientes:

$ns at 4.7 "$node(1) setdest 6 10 1"
$ns at 4.8 "$node(2) setdest 22 8 5.5"

Estos comandos permiten el movimiento de los nodos 1 y 2. Al finalizar esta acción el nodo 2 se encontrará fuera del rango de cobertura y perderá toda transmisión pendiente con el nodo 0 como podemos apreciar en la siguiente figura:



Se recomienda modificar las posiciones finales de los nodos 1 y 2 para verificar si al mantenerse en el área de cobertura la transmisión de información entre estos nodos ocurre sin ningún problema.

Al finalizar la simulación conteste las siguientes preguntas:

- **¿En que momento el nodo 2 deja de recibir paquetes del nodo 0?** *5.845 segundos*
- **Indique las posiciones iniciales y finales de los nodos 1 y 2.** Nodo 1: 1,1 Nodo 2: 15,1 Nodo 1: 6,10 Nodo 2: 22,8

5.2 Simular dos piconets conectadas entre ellas por un nodo PMP.

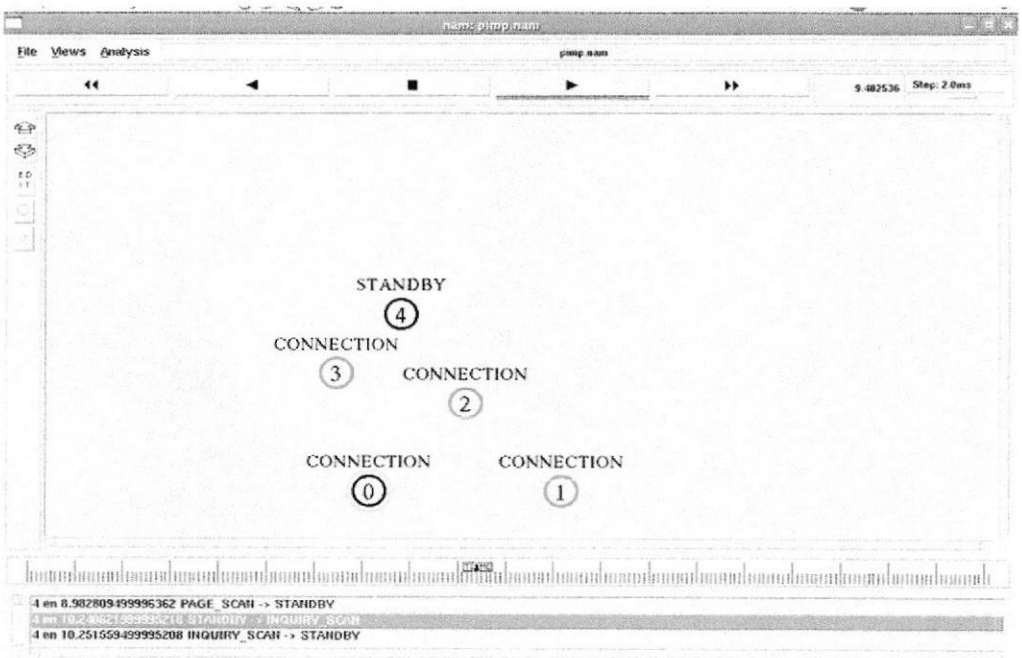
Los nodos PMPs, o también conocidos como nodos participantes en múltiples piconets, son aquellos que establecieron conexión con más de una piconet a la vez. Las razones por las cuáles estos nodos llegaron a participar en varias piconets puede ser: por haber formado parte de un procedimiento de cambio de roles, por haber aceptado aplicaciones de dispositivos que están formando piconets diferentes, o por petición de una topología compleja de red Bluetooth, como una scatternet, haciendo que este dispositivo sirva de nodo puente entre piconets por su situación geográfica.

Una vez que tengamos la pantalla del terminal, ejecutaremos las siguientes sentencias:

- cd PracticasBT
- cd Practica2
- ns pmp.tcl >> pmp.out

Al presionar play en esta simulación tendremos la formación una piconet inicial entre 0, 1, 2 y 3. Los comandos a utilizar son los siguientes:

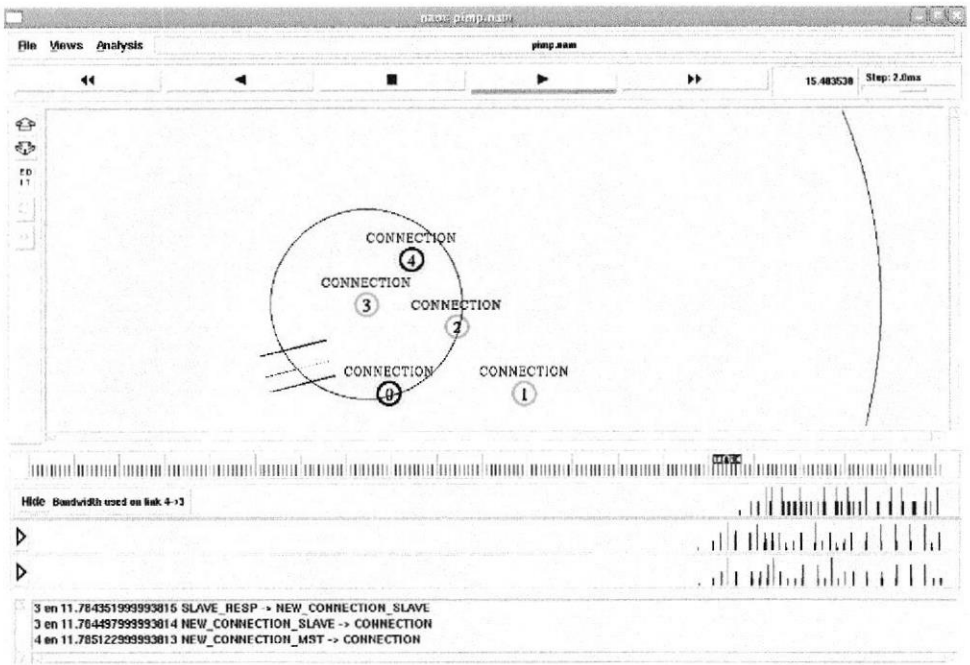
$ns at 1.2 "$node(0) make-bnep-connection $node(1) DH5 DH3 none"
$ns at 3.4 "$node(0) make-bnep-connection $node(2) DH5 DH3 none"
$ns at 5.6 "$node(0) make-bnep-connection $node(3) DH5 DH3 none"
$ns at 14.5 "$ftp0 start"



Para conectar el nodo 4 con la piconet vamos a utilizar el nodo 3 como puente con el siguiente comando:

$ns at 10.5 "$node(4) make-br $node(3) DH5 DH3 none"

Con esta sentencia creamos una nueva piconet entre el nodo 3 y el nodo 4 que a su vez se encuentra conectada con la piconet anterior gracias al nodo 3 que actúa como bridge o puente entre ambas piconets. En la siguiente figura se muestra una transferencia de archivos entre las dos piconets:



Se recomienda verificar la función del nodo puente escogiendo a otro nodo para que forme la nueva piconet con el nodo cuatro. Recuerde no escoger los nodos que participen en la transferencia de información entre las dos piconets.

Al finalizar la simulación conteste las siguientes preguntas:

- **¿Qué tipo de funciones realiza el nodo PMP?**
  Realiza la función de "puente" entre piconets que quieran establecer comunicación entre ellas. Puede adoptar las siguientes combinaciones: maestro-esclavo y esclavo-esclavo.
- **¿En qué momento se establece la piconet entre los nodos 3 y 4?**
  11,79 segundos de la simulación
- **¿Qué rol desempeña el nodo PMP en esta simulación?**
  Permite la comunicación entre las dos piconets y participa como esclavo en ambas.

5.3 Simular el intercambio de roles entre dos dispositivos conectados en una piconet.

Este procedimiento es utilizado únicamente por dos dispositivos que estén conectados en una piconet, es decir, ambos dispositivos intercambiarán roles, dando como resultado que el dispositivo maestro se convierta en el esclavo y que el esclavo se convierta en maestro. Cuando ocurre este procedimiento se forma nuevamente la piconet, en otras palabras, el esquema de saltos entre canales se redefine según el nuevo dispositivo maestro.

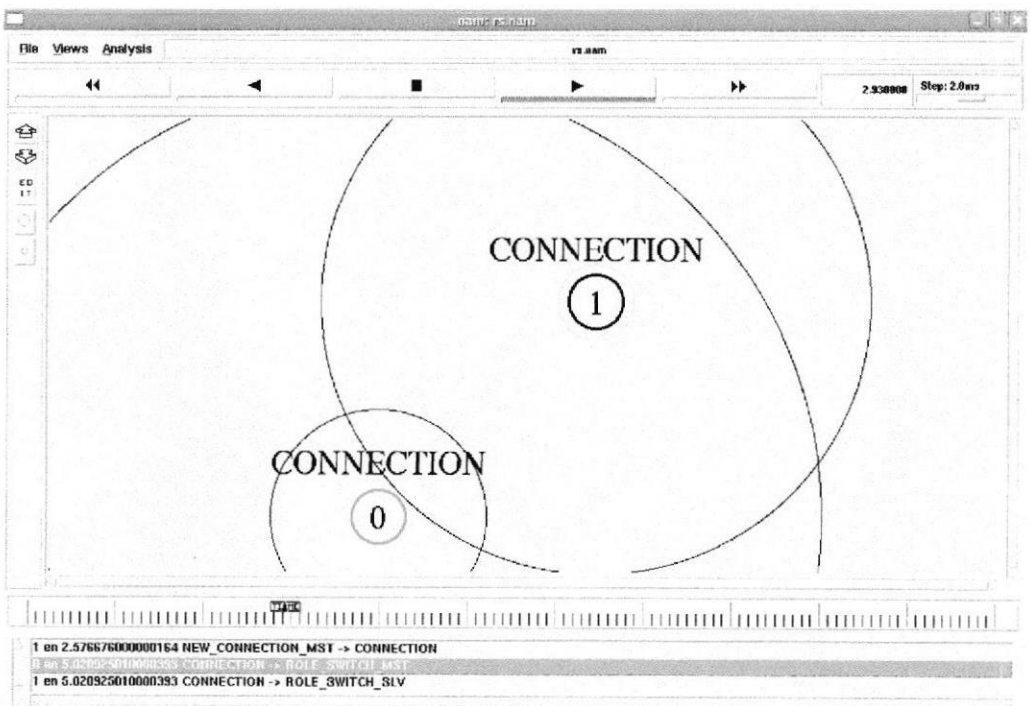
Si existiera el caso en el que uno de los dispositivos que va a realizar el cambio de roles tuviera alguna conexión con otro dispositivo, este enlace no se pierde y el dispositivo quedaría trabajando como un nodo participante en múltiples piconets.

Una vez que tengamos la pantalla del terminal, ejecutaremos las siguientes sentencias:

- cd PracticasBT
- cd Practica2
- ns rs.tcl >> rs.out

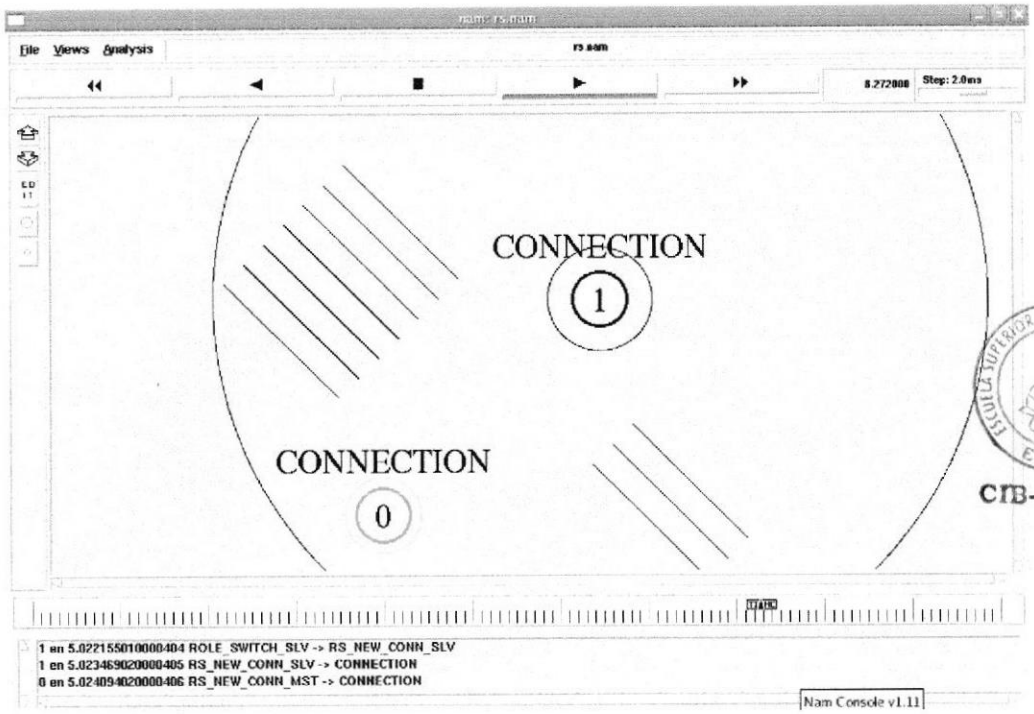
Al ejecutar esta simulación tendremos una piconet entre los nodos 1 y 0, siendo maestro y esclavo respectivamente, utilizando el siguiente comando:

$ns at 0.1 "$node(1) make-bnep-connection $node(0) DH1 DH1"



Una vez establecida la piconet procedemos al intercambio de roles entre el nodo 0 y el nodo 1, así también se establecerá una transferencia TCP entre

ambos nodos teniendo ahora como nuevo maestro al nodo 0 y como nuevo esclavo al nodo 1 como lo observamos en la siguiente figura:



Al finalizar la simulación conteste las siguientes preguntas:

- Verificar los canales de frecuencia que se utilizan en la piconet antes de que exista el intercambio de roles.
- Verificar todos los estados y paquetes intercambiados en el momento del intercambio de roles.
- Verificar los canales de frecuencia que usa la nueva piconet. ¿Existe alguna diferencia?
- Verifique el CLK que utiliza la nueva piconet.

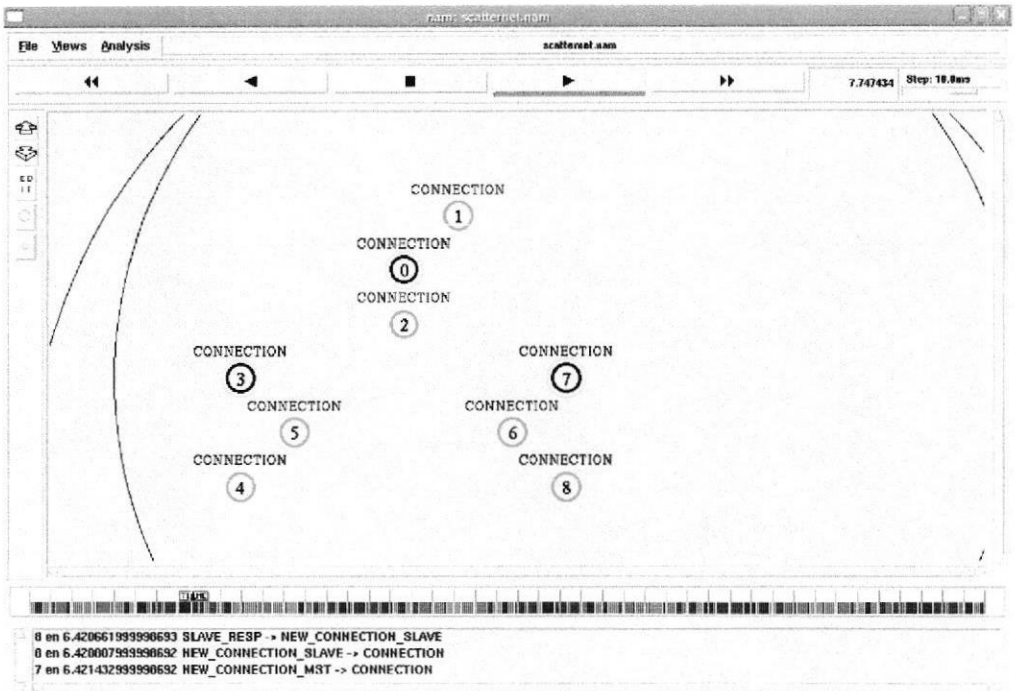
5.4 Simular una red Bluetooth avanzada (Scatternet).

Las scatternets básicamente resultan de la interconexión de 2 o más piconets. Es una topología de red bastante compleja, ya que se pueden solapar ciertos canales e incurrir a niveles de interferencias considerables entre las piconets que se interconecten, así como problemas en la transmisión de datos, ancho de banda y sincronización de los dispositivos que participan en la scatternet. Además, los nodos que enlazan estas piconets, conocidos como nodos puentes, deben participar en ambas piconets según el rol que tengan en cada una.
Una vez que tengamos la pantalla del terminal, ejecutaremos las siguientes sentencias:

- cd PracticasBT
- cd Practica2
- ns scatternet.tcl >> scatternet.out

Al ejecutar esta simulación estableceremos tres piconets entre los nodos 0, 1 y 2 la primera, 3, 4 y 5 la segunda y 6, 7 y 8 la tercera con los siguientes comandos:

$ns at 1.2 "$node(0) make-bnep-connection $node(1) DH5 DH3 none"
$ns at 1.4 "$node(0) make-bnep-connection $node(2) DH5 DH3 none"
$ns at 2.2 "$node(3) make-bnep-connection $node(4) DH5 DH3 none"
$ns at 2.4 "$node(3) make-bnep-connection $node(5) DH5 DH3 none"
$ns at 3.2 "$node(7) make-bnep-connection $node(6) DH5 DH3 none"
$ns at 3.4 "$node(7) make-bnep-connection $node(8) DH5 DH3 none"



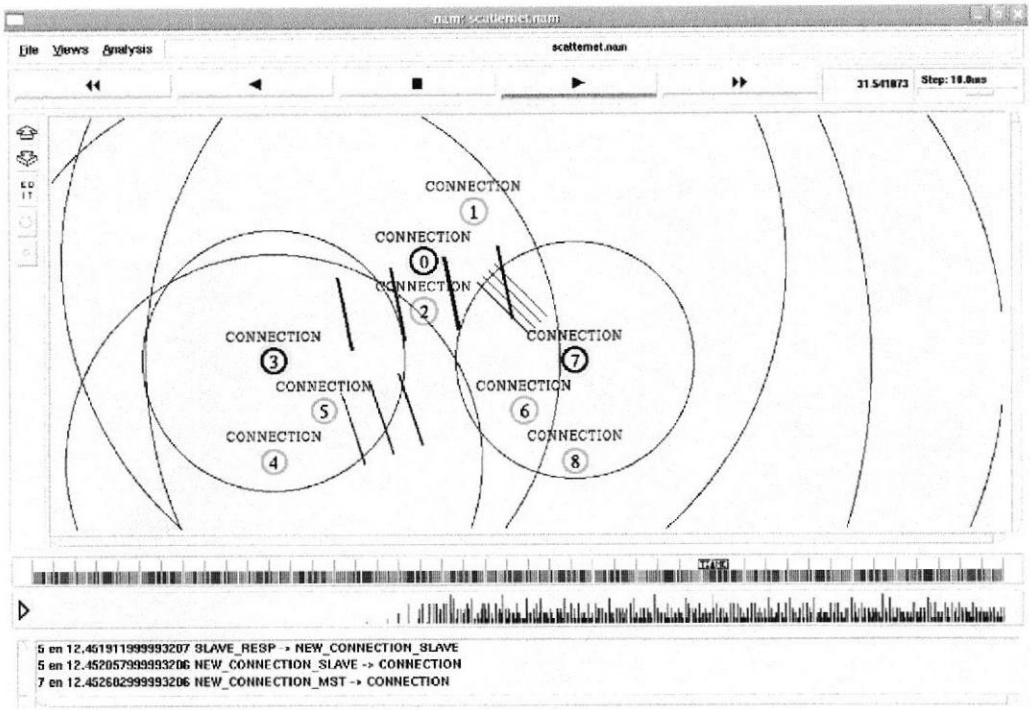
Luego procederemos a interconectar las tres piconets mediante dos nodos puente con el siguiente comando:

$ns at 10.0 "$node(3) make-br $node(2) DH5 DH3 none"
$ns at 11.0 "$node(7) make-br $node(5) DH5 DH3 none"

Finalizaremos con una transferencia de datos entre los nodos 0 y 7 para verificar que las piconets se encuentren interconectadas por medio del siguiente comando:

$ns at 15.0 "$ftp0 start"

En la siguiente figura vemos la transferencia de datos entre ambos nodos:



Se recomienda revisar el archivo tcl de esta simulación para modificar los nodos que participan en la transferencia de paquetes entre las tres piconets. Además verifique si los nodos puente siguen participando de este nuevo escenario.

BIBLIOGRAFIA

1. The VINT Project, The ns Manual, UC Berkeley, LBL, USC/ISI, and Xerox PARC,.Kevin Fall (kfall@ee.lbl.gov), Kannan Varadhan kannan@catarina.usc.edu), Editores; Enero 8, 2003.

2. BluetoothSIG, Specifications Documents, http://www.bluetooth.com/Bluetooth/Learn/Technology/Specifications/ revisada el 4 de Enero 2006.

3. D. Agrawal, Q. Wang, UCBT Bluetooth Extension for NS2 at the University of Cincinnati, http://www.ececs.uc.edu/~cdmc/ucbt/ucbt.html revisada el 15 de Enero de 2006.

APÉNDICE D.

Perfiles definidos hasta el momento en la especificación Bluetooth.

Entre los perfiles definidos hasta el momento tenemos:

- **A2DP:** Describe la manera en la cual audio de calidad estéreo es transmitido desde una fuente multimedia hacia un receptor.

- **AVCTP:** Describe los mecanismos de transporte necesarios para intercambiar mensajes en lo que respecta al control de dispositivos de audio y video.

- **AVDTP:** Describe los procedimientos de negociación, establecimiento y transmisión de audio y video.

- **AVRCP:** Diseñado para proveer una interfaz estándar en lo que se refiere al control de TVs, equipos Hi - Fi, u otros por medio de un control remoto capaz de controlar todo el equipo de audio y video al cual el usuario tiene acceso.

- **BIP:** Define como un dispositivo capaz de manejar imágenes puede ser controlado de forma remota.

- **BBP:** Permite a dispositivos enviar texto, e - mails, vCards o imágenes a impresoras listas para trabajar.

- **BNEP:** Permite a los paquetes IP ser transportados dentro de la carga útil de los paquetes L2CAP. BNEP es usado por el perfil PAN.

- **CIP:** Define cómo la señalización ISDN puede ser trasmitida a través de una conexión inalámbrica Bluetooth.

- **CTP:** Define como un teléfono inalámbrico puede ser implementado en un enlace inalámbrico Bluetooth.

- **DUN**: Provee un estándar para acceder al Internet y otros servicios dial - up por medio de tecnología Bluetooth.

- **ESDP:** Define como los dispositivos universales "plug and play" se conectan a través de la tecnología inalámbrica Bluetooth.

- **FAX:** Define como un dispositivo FAX gateway puede ser usado por un terminal.

- **FTP:** Define como las carpetas y archivos en un dispositivo servidor pueden ser examinadas desde un dispositivo cliente.

- **GAP:** Provee la base para todos los otros perfiles y define medios consistentes para establecer un enlace de banda base entre dispositivos Bluetooth conectables. GAP asegura un alto grado de interoperabilidad entre aplicaciones y dispositivos.

- **GAVDP:** Provee las bases para el funcionamiento de los perfiles A2DP y VDP.

- **GOEP:** Usado para transmitir un objeto de un dispositivo a otro. El objeto puede ser una imagen, un documento, una tarjeta de negocios, etc.

- **HFP:** Describe como un dispositivo gateway puede ser usado para realizar y recibir llamadas desde un dispositivo "manos libres".

- **HCRP:** Permite a dispositivos como impresoras y scanners conectarse con dispositivos como laptops y PCs de escritorio sin la necesidad de un cable físico.

- **HSP:** Describe como un auricular bluetooth se debería comunicar con una PC o con otro dispositivo Bluetooth como un teléfono móvil.

- **HID:** Define los protocolos, procedimientos y características que serán usados por dispositivos HID tales como teclados, apuntadores, controles de juegos y de monitoreo.

- **ICP:** Define como dos teléfonos móviles Bluetooth en la misma red se puedan comunicar directamente sin usar la red telefónica pública.

- **OBEX:** Protocolo de intercambio de datos diseñado por IrDA que tiene como fin abstraer y normalizar la forma en que se comunican los pequeños dispositivos electrónicos que van surgiendo con necesidad de comunicarse con otros.

- **OPP:** Define los roles de empuje del servidor y del cliente.

- **PAN:** Describe cómo dos o más dispositivos Bluetooth pueden formar una red Ad - Hoc y como el mismo mecanismo puede ser usado para acceder a una red remota a través de un punto de acceso.

- **RFCOMM:** Proporciona emulación de puertos seriales a través del protocolo L2CAP. Este protocolo se basa en el estándar de la ETSI denominado TS 07.10. RFCOMM es un protocolo de transporte sencillo, con soporte para hasta 9 puertos seriales RS - 232. El protocolo RFCOMM permite hasta 60 conexiones simultáneas entre dos dispositivos Bluetooth.

- **SDP:** Permite a las aplicaciones cliente descubrir la existencia de diversos servicios proporcionados por uno o varios servidores de aplicaciones, junto con los atributos y propiedades de los servicios que se ofrecen. Estos atributos de servicio incluyen el tipo o clase de servicio ofrecido y el mecanismo o la información necesaria para utilizar dichos servicios.

- **SDAP:** Describe como una aplicación debería utilizar el protocolo SDP para descubrir servicios en un dispositivo remoto.

- **SAP:** Permite a dispositivos como teléfonos de automóviles con transceivers GSM conectarse a una tarjeta SIM de un teléfono Bluetooth.

- **SPP:** Permite que dispositivos Bluetooth realicen simulación de RS232. El escenario cubierto por este perfil trata con aplicaciones comerciales que utilizan Bluetooth como un substituto del cable, utilizando una capa de abstracción que representa un puerto serie virtual.

- **SYNC:** Usado en conjunto con GOEP para habilitar la sincronización del calendario y la información de direcciones entre dos dispositivos Bluetooth.

- **TCS - Binario o TCP:** Define como un dispositivo Bluetooth puede ser usado como teléfono inalámbrico.

- **VDP:** Define como un dispositivo Bluetooth transmite video a través de tecnología inalámbrica Bluetooth.

- **WAP:** Define como el protocolo de aplicación inalámbrica puede funcionar a través de un enlace de tecnología inalámbrica Bluetooth.

BIBLIOGRAFIA

1. The VINT Project, The ns Manual, UC Berkeley, LBL. USC/ISI, and Xerox PARC, Kevin Fall (kfall@ee.lbl.gov), Kannan Varadhan (kannan@catarina.usc.edu), Editores; Enero 8, 2003, página 19.

2. The VINT Project, The ns Manual, UC Berkeley, LBL, USC/ISI, and Xerox PARC, Kevin Fall (kfall@ee.lbl.gov), Kannan Varadhan (kannan@catarina.usc.edu), Editores; Enero 8, 2003, página 353.

3. LIP6, 38.1.11 Nam Trace File Format Lookup Table. http://www-rp.lip6.fr/~ridoux/Docs/Manuel-NS2/node514.html revisada el 1 de Agosto de 2006.

4. D. Agrawal, Q. Wang, UCBT Bluetooth Extension for NS2 at the University of Cincinnati, http://www.ececs.uc.edu/~cdmc/ucbt/ucbt.html revisada el 15 de Enero de 2006.

5. BluetoothSIG, Specifications Documents, http://www.bluetooth.com/Bluetooth/Learn/Technology/Specifications/ revisada el 4 de Enero de 2006

6. PaloWireless, Bluetooth Resource Center, http://www.palowireless.com/infotooth/tutorial.asp revisada el 10 de Enero de 2006

7. Universidad del Sur de California, Information Science Institute, Network Simulator 2, http://www.isi.edu/nsnam/ns/ revisada el 8 de Enero de 2006

8. Instituto Tecnológico de Informática, Valencia - España, Redes Inalámbricas Ad Hoc Pietro Masón, Juan Carlos Pietro, http://www.iti.upv.es/services/reviewtic/public/2006/03/pdf/articulo_redes.pdf/attach/articulo_redes.pdf página 5, revisado el 20 de Septiembre de 2006

9. M. I. T. Computer Science and Artificial Intelligence Laboratory - USA, Blueware: Bluetooth Simulator for ns, http://nms.csail.mit.edu/projects/blueware/software/ revisada el 10 de Enero de 2006

10. IBM, Open source Projects and Resources, http://www-128.ibm.com/developerworks/opensource/ revisada el 10 de Enero de 2006

11. ICE Universidad de Lleida, Manual de Linux, Álvaro Alea, http://www.ice.udl.es/udv/manuals/linux.pdf revisada el 8 Enero de 2006

12. Universidad del Valle Cali-Colombia, Implementación de una red inalámbrica bluetooth, O. Rodríguez, R. Maya, www.univalle.edu.co/~telecomunicaciones/trabajos_de_grado/informes/tg_OscarRodriguez_RicardoMaya.pdf revisado el 5 de Enero de 2006