

Arroyo Christian

Contenido

[Network Simulator. 3](#_Toc458588148)

[Introducción 3](#_Toc458588149)

[Marco Teórico 3](#_Toc458588150)

[Ubuntu 3](#_Toc458588151)

[Descripción Interna NS 3](#_Toc458588152)

[Network Component object 4](#_Toc458588153)

[Network Setup Helping Module 5](#_Toc458588154)

[INSTALACIÓN UBUNTU 6](#_Toc458588155)

[INSTALACIÓN NS-2 en UBUNTU 10](#_Toc458588156)

[Explicación del funcionamiento de la red en NS2 15](#_Toc458588157)

[Conclusiones 15](#_Toc458588158)

[Recomendaciones 15](#_Toc458588159)

[Bibliografía 15](#_Toc458588160)

# Network Simulator.

## Introducción

A través de los años se ha hecho importante el modelamiento de diversos eventos antes de tomar decisiones. Tal es el caso de la programación lineal, donde se plantea un problema de la vida cotidiana y mediante un modelamiento matemático es factible encontrar una o más soluciones. Sin embargo, en la realidad hay que considerar miles de factores importantes que lamentablemente un modelo matemático no considera en el mayor de los casos, pero aproxima bastante a una solución que nos puede llevar por un buen camino. Es por ello que la idea de modelar y/o simular es bastante importante en la toma de decisiones. En el presente trabajo se dará una descripción a lo que es este simulador de redes. Esta investigación está orientada a conocer el tema de la simulación de redes mediante esta aplicación denominada Network Simulator. Se realizarán pruebas de algunos algoritmos ya realizados y se hará una descripción interna de cómo funciona este simulador. Este simulador fue probado bajo una plataforma Ubuntu 12.04. Este simulador también funciona bajo plataforma Windows, sin embargo, la investigación no se llevó a cabo en este sistema operativo, por lo tanto, requisitos, funcionamiento, u otros, no fueron probados. Por lo tanto, al hablar de línea de comandos se hará referencia a las consolas de Linux.

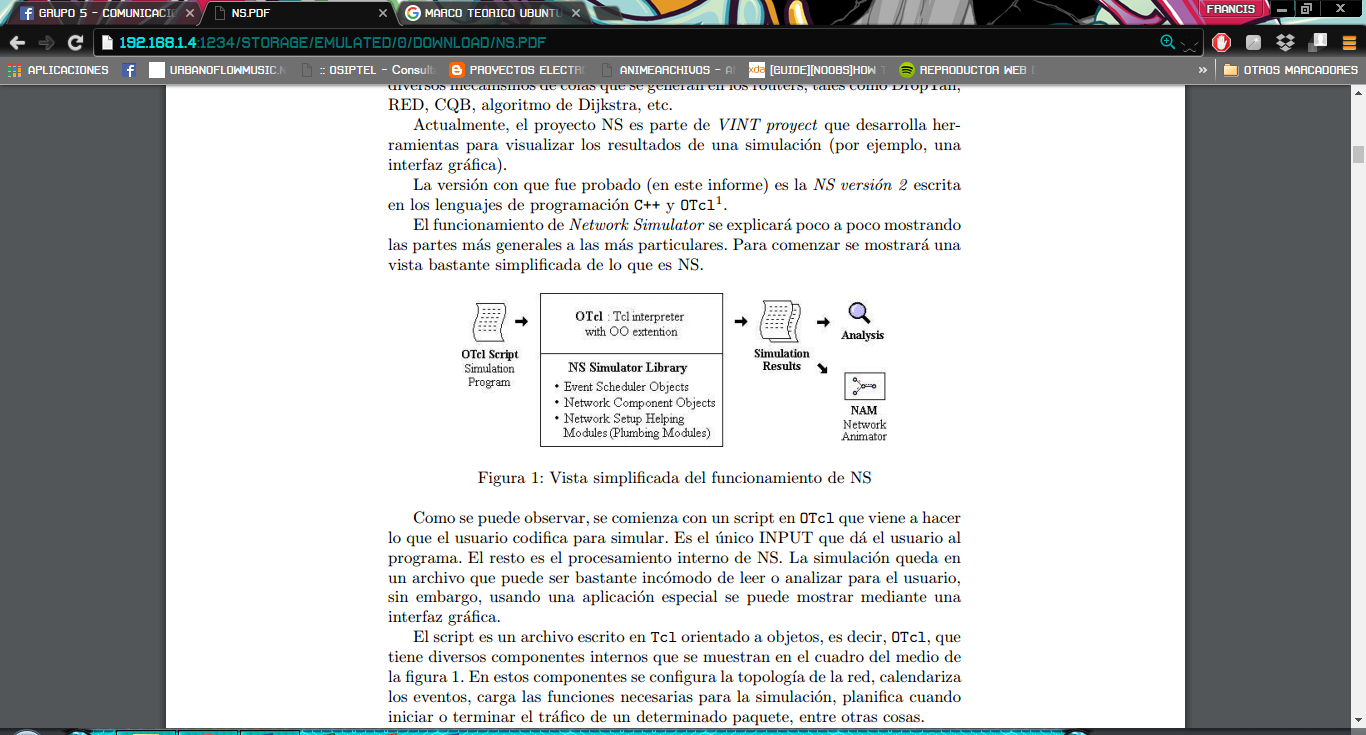
## Marco Teórico

### Ubuntu

Ubuntu es un sistema operativo basado en Debian y que se distribuye como software libre y gratuito, el cual incluye su propio entorno de escritorio denominado Unity. Está orientado al usuario novel y promedio, con un fuerte enfoque en la facilidad de uso y en mejorar la experiencia de usuario. Está compuesto de múltiple software normalmente distribuido bajo una licencia libre o de código abierto. Estadísticas web sugieren que la cuota de mercado de Ubuntu dentro de las "distribuciones Linux" es, aproximadamente, del 49%, y con una tendencia a aumentar como servidor web. Y un importante incremento activo de 20 millones de usuarios para fines del 2011.

### Descripción Interna NS

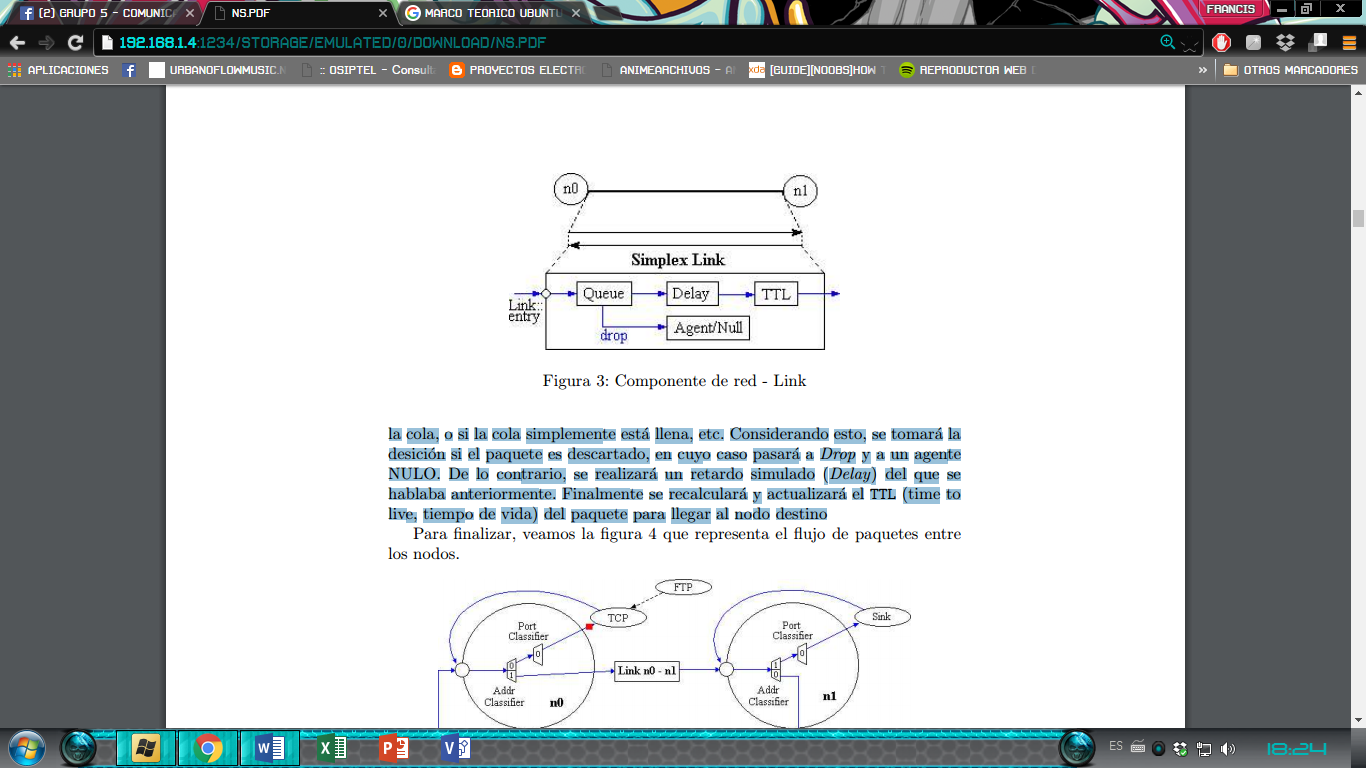
Network Simulator es un simulador discreto de eventos creado por la Universidad de Berkeley para modelar redes de tipo IP. En la simulación se toma en cuenta lo que es la estructura (topología) de la red y el tráfico de paquetes que posee la misma, con el fin de crear una especie de diagnóstico que nos muestre el comportamiento que se obtiene al tener una red con ciertas características. Trae implementaciones de protocolos tales como TCP y UDP, que es posible hacerlos comportar como un tráfico FTP, Telnet, Web, CBR y VBR. Maneja diversos mecanismos de colas que se generan en los routers, tales como DropTail, RED, CQB, algoritmo de Dijkstra, etc. Actualmente, el proyecto NS es parte de VINT proyect que desarrolla herramientas para visualizar los resultados de una simulación (por ejemplo, una interfaz gráfica). La versión con que fue probado (en este informe) es la NS versión 2 escrita en los lenguajes de programación C++ y OTcl1. El funcionamiento de Network Simulator se explicará poco a poco mostrando las partes más generales a las más particulares. Para comenzar se mostrará una vista bastante simplificada de lo que es NS.



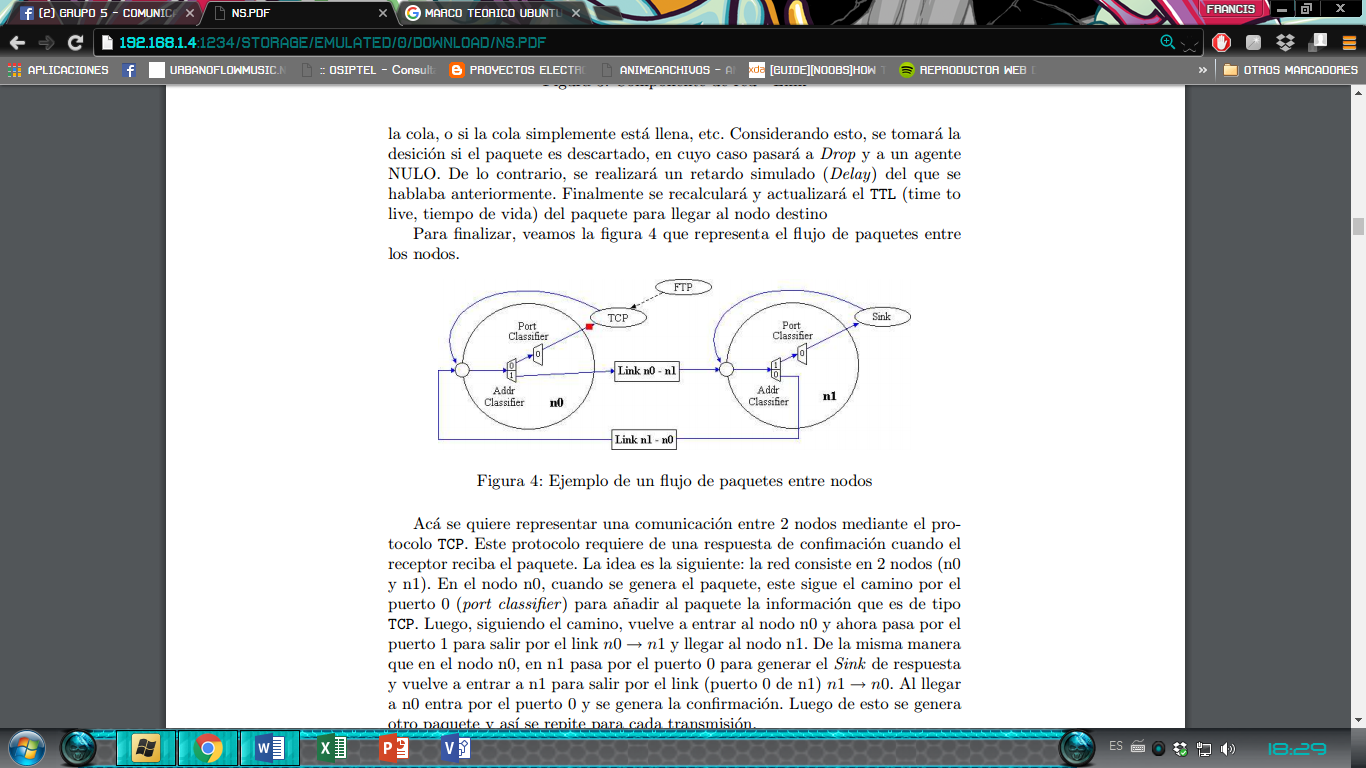
Como se puede observar, se comienza con un script en OTcl que viene a hacer lo que el usuario codifica para simular. Es el único INPUT que da el usuario al programa. El resto es el procesamiento interno de NS. La simulación queda en un archivo que puede ser bastante incomodo de leer o analizar para el usuario, sin embargo, usando una aplicación especial se puede mostrar mediante una interfaz gráfica. El script es un archivo escrito en Tcl orientado a objetos, es decir, OTcl, que tiene diversos componentes internos que se muestran en el cuadro del medio de la figura 1. En estos componentes se configura la topología de la red, calendariza los eventos, carga las funciones necesarias para la simulación, planifica cuando iniciar o terminar el tráfico de un determinado paquete, entre otras cosas. A continuación, se entrará a especificar un poco más como funciona cada componente. Tampoco es la idea de entrar en detalle, sin embargo, una referencia rápida a cada punto será justa y necesaria.

### Network Component object

Se encarga de hacer consistente la comunicación que hay entre distintos componentes de red, por donde pasarán los paquetes. Los componentes de red pueden ser; el ancho de banda de un link, un link unidireccional o bidireccional, retardos de paquetes, etc. En el caso de los retardos también actúa´ el event scheduler. A modo de ejemplo, en la figura se muestra el componente de red que permite unir dos nodos, es decir, un link. En esta figura se representa un link simple unidireccional. En el caso de requerir uno bidireccional, simplemente se crea otro objeto con la misma estructura para el lado contrario. En la entrada al link el paquete deberá quedar en la cola. Acá se realizarán una serie de procesamientos dependiendo del tipo de cola que tenga ese link, tales como, si el tamaño del paquete supera el tamaño de la cola, o si la cola simplemente está llena, etc. Considerando esto, se tomará la decisión si el paquete es descartado, en cuyo caso pasará a Drop y a un agente NULO. De lo contrario, se realizará un retardo simulado (Delay) del que se hablaba anteriormente. Finalmente se recalculará y actualizará el TTL (time to live, tiempo de vida) del paquete para llegar al nodo destino



Para finalizar, en la siguiente figura se representa el flujo de paquetes entre los nodos.



Acá se quiere representar una comunicación entre 2 nodos mediante el protocolo TCP.

Este protocolo requiere de una respuesta de confinación cuando el receptor reciba el

paquete. La idea es la siguiente: la red consiste en 2 nodos (n0 y n1). En el nodo n0,

cuando se genera el paquete, este sigue el camino por el puerto 0 (port classifier ) para

añadir al paquete la información que es de tipo TCP. Luego, siguiendo el camino, vuelve a

entrar al nodo n0 y ahora pasa por el puerto 1 para salir por el link n0 → n1 y llegar al

nodo n1. De la misma manera que en el nodo n0, en n1 pasa por el puerto 0 para generar

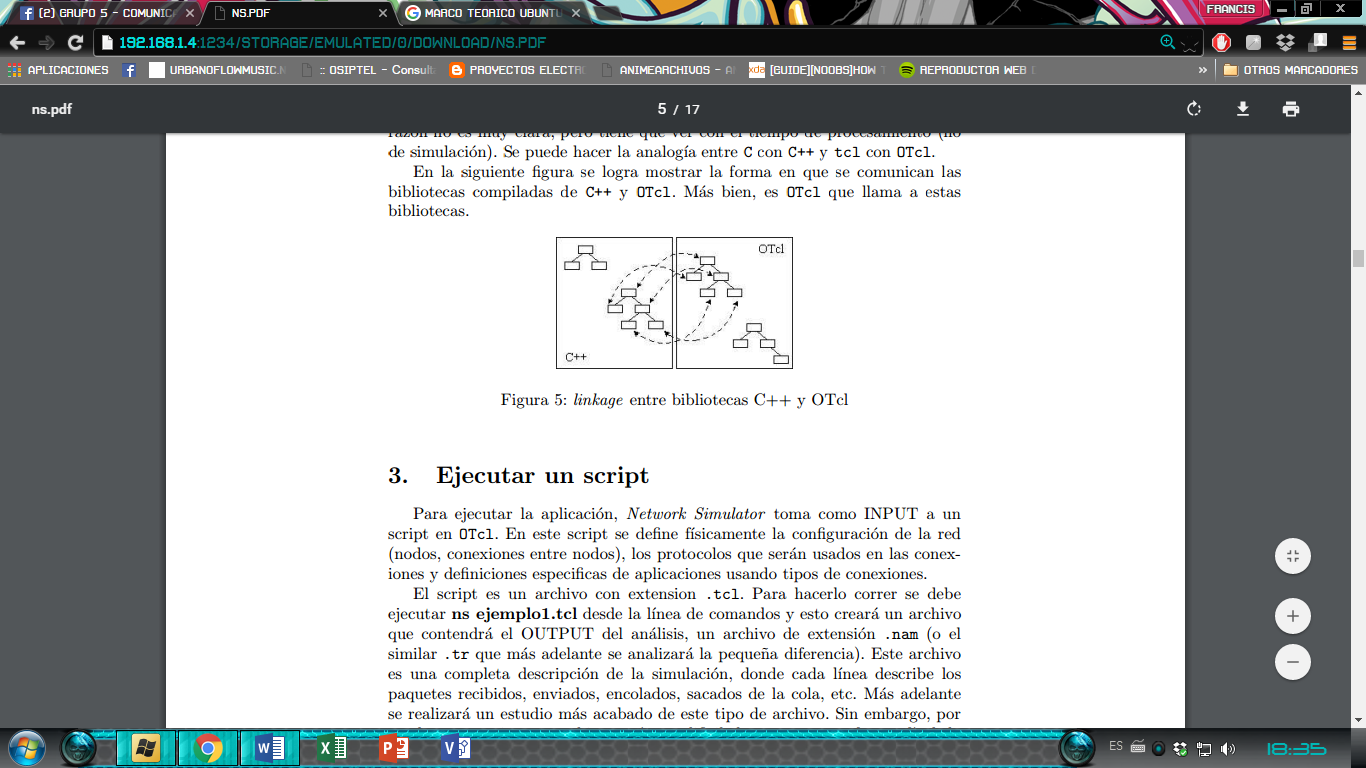
el Sink de respuesta y vuelve a entrar a n1 para salir por el link (puerto 0 de n1) n1 → n0.

Al llegar a n0 entra por el puerto 0 y se genera la confirmación. Luego de esto se genera

otro paquete y así se repite para cada transmisión.

### Network Setup Helping Module

Por último, el network Setup Helping Modules indicará las bibliotecas necesarias para realizar la simulación. Esto es necesario ya que los 2 primeros componentes, descritos en los subítemes 2.2 y 2.3, están escritos y compilados en C++ y están disponibles para el intérprete OTcl a través de un linkage 2 . La razón no es muy clara, pero tiene que ver con el tiempo de procesamiento (no de simulación). Se puede hacer la analogía entre C con C++ y tcl con OTcl. En la siguiente figura se logra mostrar la forma en que se comunican las bibliotecas compiladas de C++ y OTcl. Más bien, es OTcl que llama a estas bibliotecas.



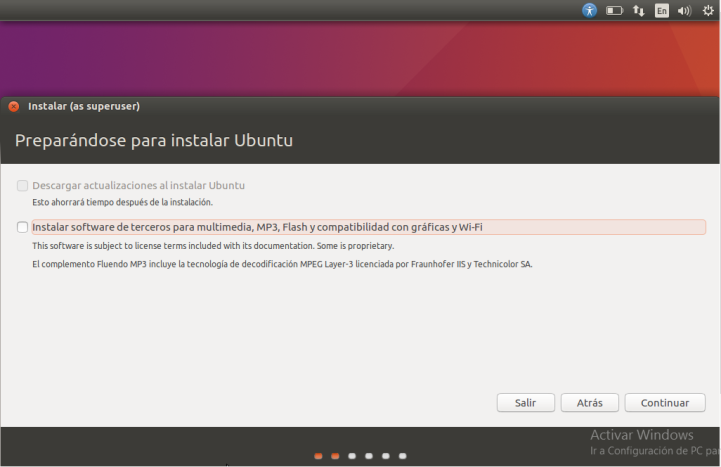
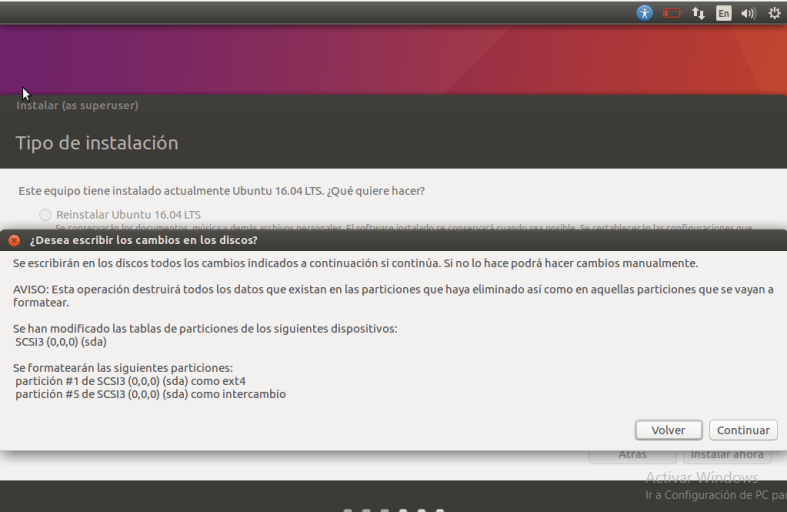
## INSTALACIÓN UBUNTU

1. Descargar el siguiente enlace. http://www.ubuntu.com/  
2. Iniciar instalación.

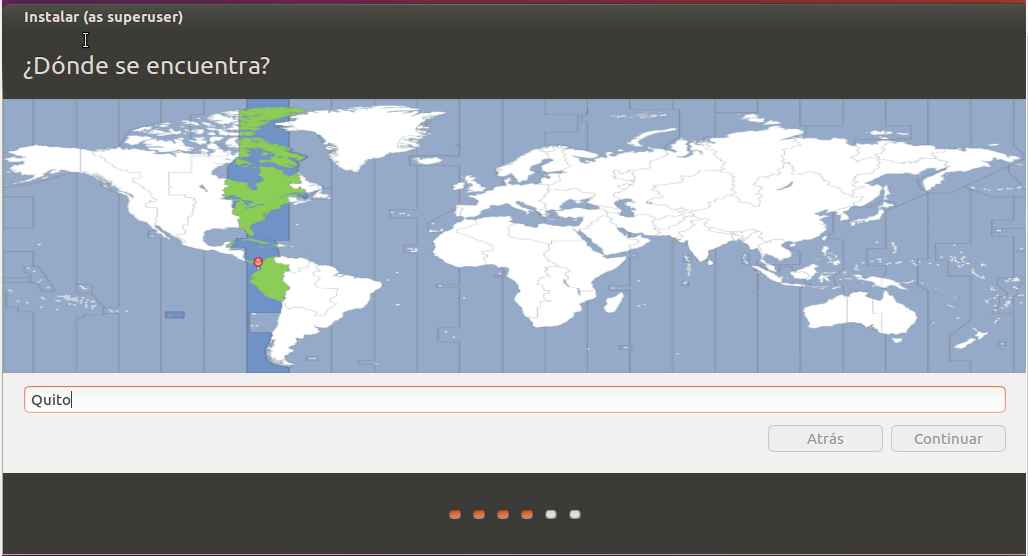


3. Seleccionar el idioma e inicializar instalación.

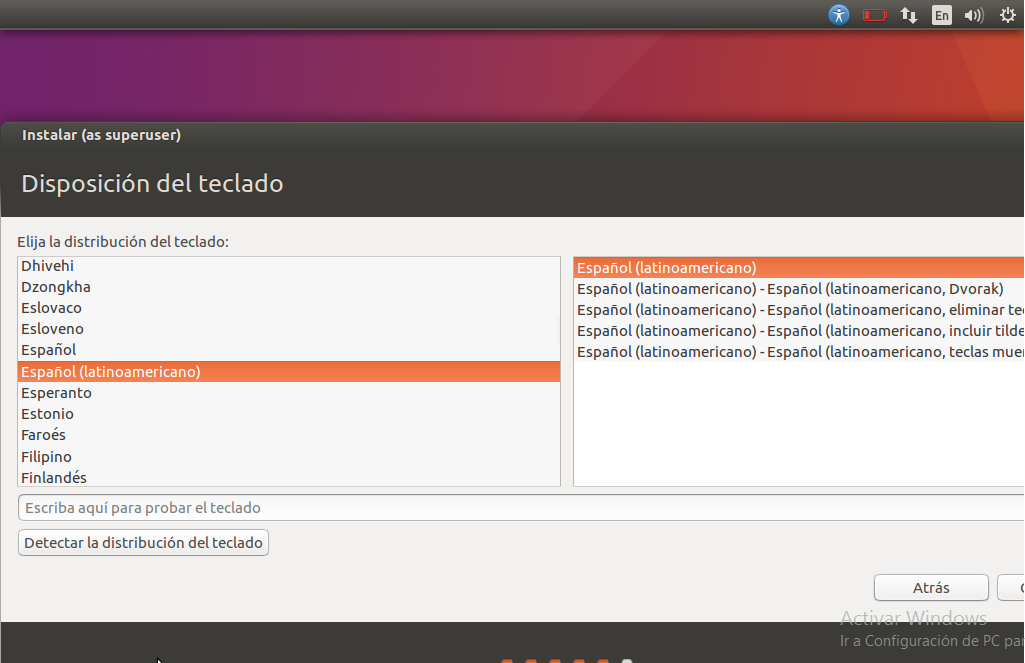
4. Seleccionar ‘CONTINUAR’.



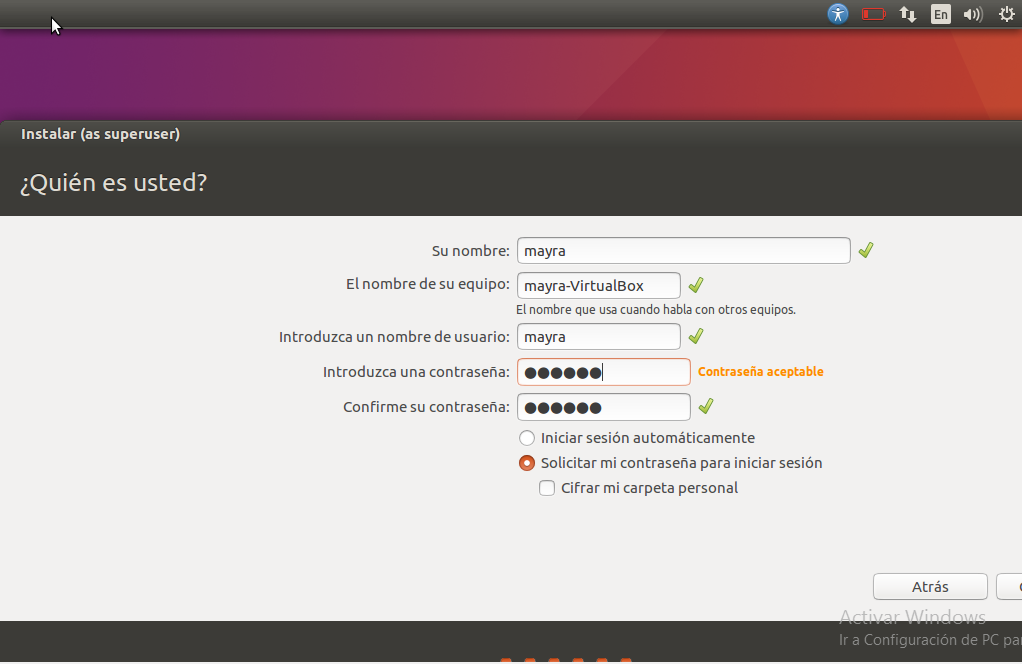
5. Colocar la cuidad donde se encuentra y seleccionar ‘CONTINUAR’



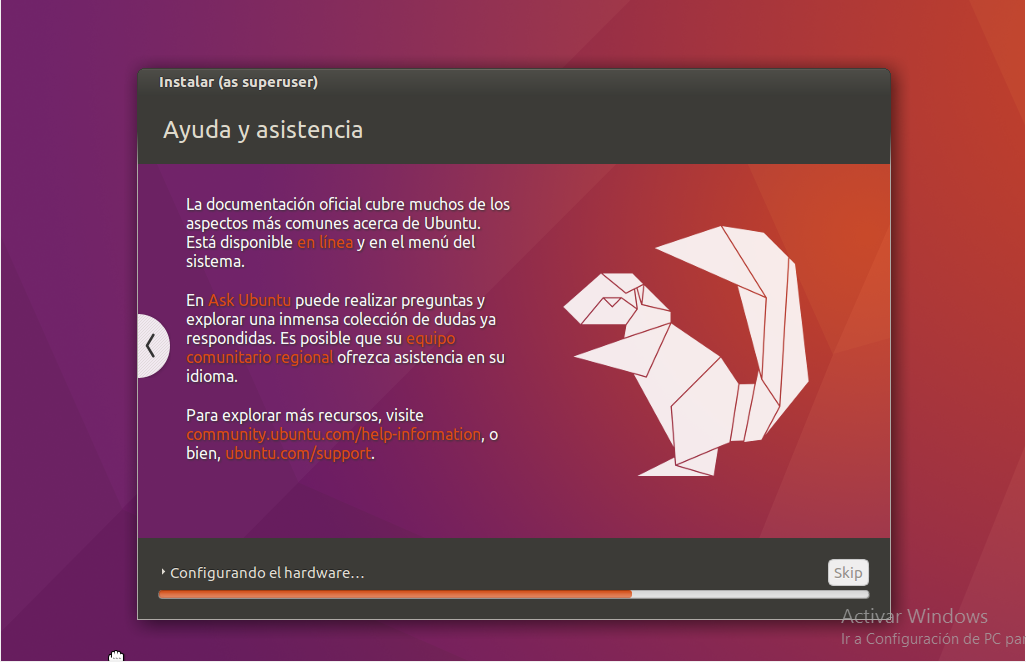
6. Selecciona disposición del teclado y ‘CONTINUAR’



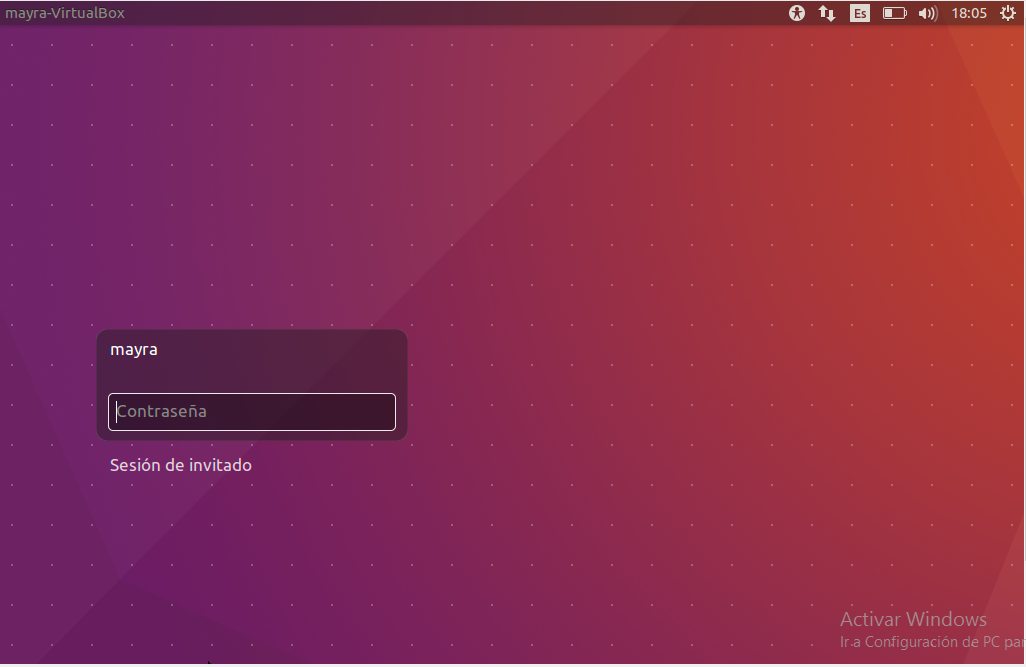
7. Coloca nombre y contraseña con la que va a iniciar sesión en UBUNTU, seleccionar ‘CONTINUAR’



8. Empieza la instalación



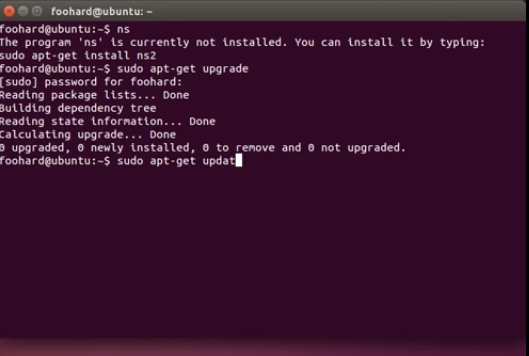
9. Coloca nombre y contraseña para iniciar sesión en UBUNTU.



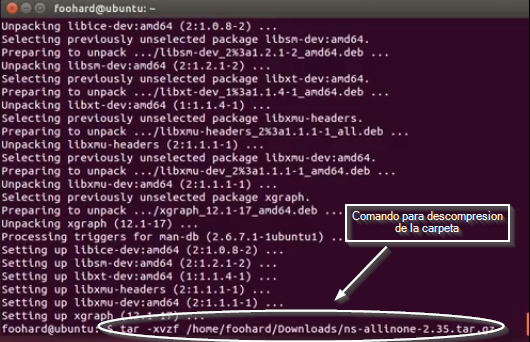
## INSTALACIÓN NS-2 en UBUNTU

1. Inicialmente debemos descargar los paquetes con los cuale va a funcionar la aplicación:

* Apt-get install upgrade
* Apt-get install update

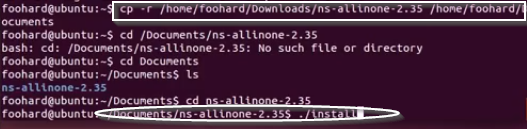


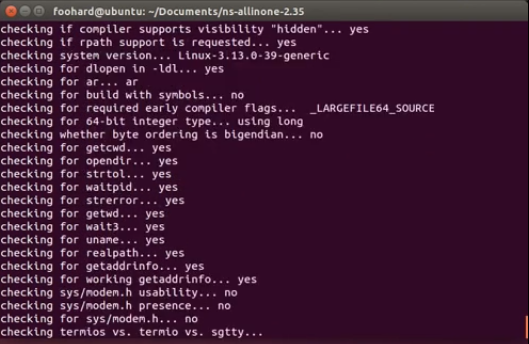
1. Debemos descargar el paquete ns-allinone2.35.tar.gz y descomprimir el mismo para su utilización.



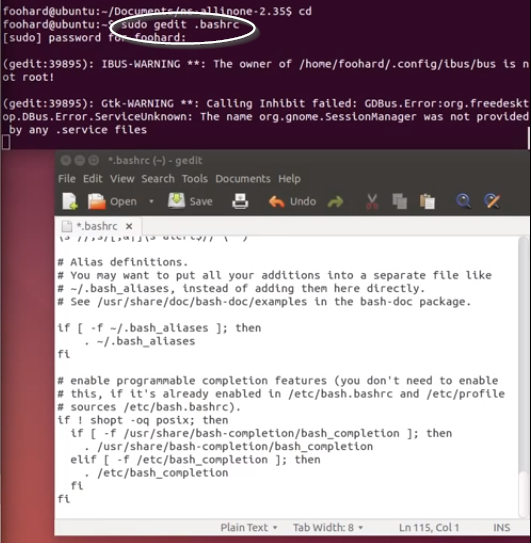
1. Una vez descomprimida la carpeta nos dirigimos entre los directorios hacia la ubicación con el comando “cd” y una vez en el directorio ingresamos en el prompt “./install”.

Esto va a dar inicio a la descarga e instalación.

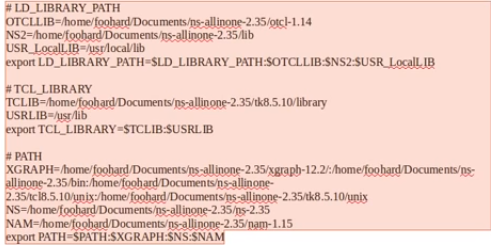


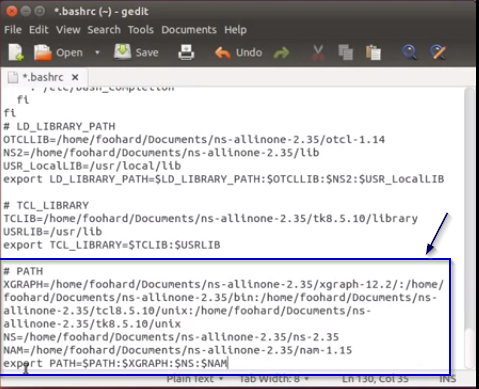


1. Es necesario modificar el archivo .bashrc para generar la path de manera correcta, ya que la ubicación de los archivos se encuetran en otra carpeta

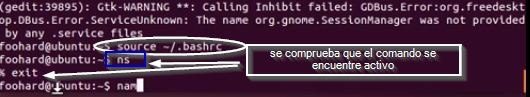


1. Al archivo .bashrc se debe añadir las siguientes PATH:

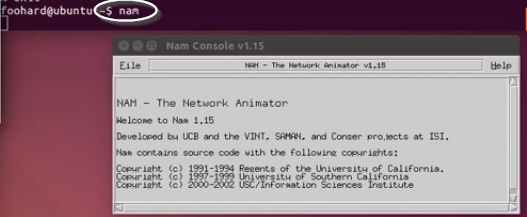


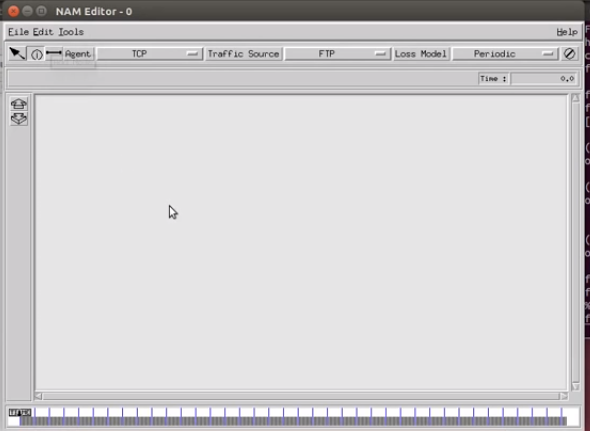


1. Se ejecuta el comando “source ./bashrc” para ejecutar los cambios ejecutados con las nuevas PATH.



1. Una vez comprobado n, se procede a colocar el comando nam para la utilización de la interfaz gráfica de NS-2 y empezar con su programación





## Explicación del funcionamiento de la red en NS2

Como se puede ver en la simulación existen 4 enlaces conectados a un nodo central, que haría la forma de la topología en estrella. El nodo central es el que va a poder comunicar un enlace con otro, los enlaces se los ha puesto de cuatro colores diferentes: rojo, azul, anaranjado y verde, para una mejor comprensión y apreciación.

Se ha enumerado a los enlaces del 1 al 4 y el nodo central tiene el número cero. Al hacer click en el botón de play podemos ver como evoluciona en el tiempo la conexión entre enlaces. Se puede ver como las líneas se entrecortan o se hacen más pronunciadas.

Con nuestro diagrama podemos ver que todos los enlaces tienen comunicación directa con el nodo cero pero no tienen comunicación directa entre enlaces sino sólo a través del nodo cero.

## Conclusiones

* Podemos ver que la funcionalidad de la red depende de la forma de conexión o la topología que se le haya asignado a los enlaces.
* Las líneas más pronunciadas nos dan a entender que la comunicación es mucho mejor que cuando se nos muestran líneas entrecortadas.
* El nodo central es el que está a cargo de la comunicación entre enlaces, si éste llegara a tener problemas, la red en general tendría problemas.

## Recomendaciones

* Se recomienda usar Ubuntu para la implementación de NS2 ya que en las otras plataformas no se tuvieron buenos resultados.
* Es más fácil distinguir los enlaces si a éstos se les asigna un color, y es más provechoso para distinguir alguna falla en la red.

## Bibliografía

* http://www.isi.edu/nsnam/ns/
* http://www.postel.org/pipermail/end2end-interest/2015-February/009286.html
* http://profesores.elo.utfsm.cl/~tarredondo/info/networks/ns2/intro-ns2.pdf
* https://github.com/vsubhashini/queue-ns2/blob/master/3-TCP-RED/red.tcl