**SIMULADOR INTERACTIVO DE REDES LAN**

**DE TECNOLOGÍA ETHERNET**

**MIGUEL RUIZ**

**MAURICIO VALENCIA CASTILLO**

**INGENIERO LUIS CARLOS TRUJILLO ARBOLEDA**

**Director**

****

**PONTIFICIA UNIVERSIDAD JAVERIANA**

**FACULTAD DE INGENIERÍA**

**DEPARTAMENTO DE INGENIERÍA ELECTRÓNICA**

**PROYECTO DE GRADO**

**BOGOTÁ**

**2008**

Contenido

[INTRODUCCIÓN 4](#_Toc208582001)

[2. OBJETIVO GENERAL Y OBJETIVOS ESPECÍFICOS 5](#_Toc208582002)

[2.1 OBJETIVO GENERAL 5](#_Toc208582003)

[2.2 OBJETIVOS ESPECÍFICOS 5](#_Toc208582004)

[3. ESPECIFICACIONES 5](#_Toc208582005)

[3.1 IMPLEMENTACIÓN CAPA FÍSICA. 5](#_Toc208582006)

[3.1.1 Lógica de negociación para la capa física (capa 1): 6](#_Toc208582007)

[3.1.2 Visualización de la Capa Física: 6](#_Toc208582008)

[3.2 IMPLEMENTACIÓN CAPA DE ENLACE DE DATOS (CAPA 2) 6](#_Toc208582009)

[3.2.1 Lógica de negociación para la capa de enlace de datos: 6](#_Toc208582010)

[3.2.2. Visualización de la capa de enlace de datos. 6](#_Toc208582011)

[3.3 IMPLEMENTACIÓN CAPA DE RED (CAPA 3) 7](#_Toc208582012)

[3.3.1 Lógica de negociación para la capa de red 7](#_Toc208582013)

[3.3.2 Visualización para la capa de red. 7](#_Toc208582014)

[3.4 IMPLEMENTACIÓN CAPA DE TRANSPORTE (CAPA 4) 7](#_Toc208582015)

[3.4.1 Lógica de negociación para la capa de transporte. 8](#_Toc208582016)

[3.4.2 Visualización de la capa de Transporte 8](#_Toc208582017)

[3.5 IMPLEMENTACIÓN DISPOSITIVOS: 8](#_Toc208582018)

[3.5.1 HUB 8](#_Toc208582019)

[3.5.2 SWITCH 8](#_Toc208582020)

[3.5.3 ROUTER: 8](#_Toc208582021)

[3.5.4 HOST 9](#_Toc208582022)

[3.6 ALMACENAMIENTO DE LA INFORMACIÓN DE LA TOPOLOGÍA DE RED. 9](#_Toc208582023)

[3.7 ACCESO REMOTO DE SESIONES MÚLTIPLES DE LA APLICACIÓN. 9](#_Toc208582024)

[4. MARCO CONCEPTUAL 9](#_Toc208582025)

[4.1 REDES DE DATOS LAN TECNOLOGÍA ETHERNET 10](#_Toc208582026)

[4.1.1 Capa física 12](#_Toc208582027)

[Capa de enlace 12](#_Toc208582028)

[4.1.3 Capa de red 12](#_Toc208582029)

[4.1.4 Capa de transporte 12](#_Toc208582030)

[4.1.5 Capa de sesión 13](#_Toc208582031)

[4.1.6 Capa de presentación 13](#_Toc208582032)

[4.1.7 Capa de aplicación 13](#_Toc208582033)

[4.2 TEORÍA DE OTROS SIMULADORES 13](#_Toc208582034)

[Software actual para la simulación de redes de comunicaciones 13](#_Toc208582035)

[4.3 .NET FRAMEWORK Y EL ENTORNO COMÚN DE EJECUCIÓN PARA LENGUAJES (CLR) 15](#_Toc208582036)

[5. DESCRIPCIÓN GENERAL Y DIAGRAMA EN BLOQUES 17](#_Toc208582037)

[5.1 DESARROLLO DE LA APLICACIÓN EN CAPAS: 17](#_Toc208582038)

[5.2 COMUNICACIÓN ENTRE LAS DIFERENTES CAPAS 19](#_Toc208582039)

[5.3 DIAGRAMA EN BLOQUES: 22](#_Toc208582040)

[6. DESARROLLO TEÓRICO 24](#_Toc208582041)

[6.3 CONTROL DE VERSIONES: 25](#_Toc208582042)

[7. PRUEBAS DE CALIDAD Y DESEMPEÑO 26](#_Toc208582043)

[8. COSTOS Y FUENTES DE FINANCIACIÓN 27](#_Toc208582044)

[9. JUSTIFICACIÓN DE POSIBLES DIFERENCIAS ENTRE LA PROPUESTA Y EL PROYECTO 28](#_Toc208582045)

[10. BIBLIOGRAFIA Y FUENTES DE INFORMACIÓN 30](#_Toc208582046)

[11. OBSERVACIONES 31](#_Toc208582047)

[12. ANEXOS 32](#_Toc208582048)

# INTRODUCCIÓN

La simulación de sistemas es de vital importancia en el mundo actual, casi en cualquier área de desarrollo humano se encuentran paquetes de software especializados en emular sistemas reales, para que estos puedan ser analizados e investigados, sin necesidad de interactuar directamente con el sistema real.

El presente trabajo de grado proyecta realizar el diseño de un software de simulación de redes LAN con tecnología Ethernet, para ser usado por estudiantes del área de comunicaciones de la carrera de Ingeniería Electrónica. La característica más especial del simulador es ser un sistema multiusuario, donde varios computadores interconectados por internet o por un Router, podrán compartir el mismo entorno de simulación de modo que los estudiantes pueda interactuar entre ellos analizando y aprendiendo de los aspectos más importantes de una red de datos Ethernet. Este software funcionará de forma didáctica de modo que el aprendizaje de la gran mayoría de las características que describen a una red de datos sea conciso y entendible de forma rápida y sencilla.

En el mercado existen diferentes paquetes de simulación para el análisis de redes, cuyas características no satisfacen nuestras necesidades, además estos no están enfocados directamente al contexto académico. Gracias al desarrollo del presente proyecto de grado se tendrá un simulador especializado en el ámbito académico, con las características y funciones de acuerdo a las necesidades de los estudiantes del área de comunicaciones de la carrera de Ingeniería Electrónica

El software que desarrollaremos será una aproximación a un escenario de red real, donde se encontrarán dispositivos generalmente usados en una red LAN Ethernet, estos dispositivos se podrán interconectar libremente, posibilitando el análisis de su topología y características.

Es importante resaltar que gracias al aporte de otros estudiantes y personas interesadas en la actualización del simulador, se lograría que el simulador que inicialmente tendrá funciones básicas, en un futuro pueda acercarse muy detalladamente a un escenario real incluyendo diferentes dispositivos y protocolos.

# 2. OBJETIVO GENERAL Y OBJETIVOS ESPECÍFICOS

El proyecto que se propone desarrollar, pretende ser un complemento de gran apoyo al Área de Comunicaciones cumpliendo los siguientes objetivos y características.

## 2.1 OBJETIVO GENERAL

Desarrollar una plataforma de software Multiusuario que permita simular redes de datos tipo LAN con tecnología Ethernet. Este software será una herramienta de apoyo para el Área de comunicaciones de la carrera de Ingeniería Electrónica.

## 2.2 OBJETIVOS ESPECÍFICOS

2.2.1 Desarrollar La lógica de Negocio y la Visualización para: Capa física, capa de enlace de dato, capa de red y capa de transporte según el modelo OSI (Open System Interconection).

2.2.2 Desarrollar la Lógica de Negocio y la Visualización para la implementación de los siguientes dispositivos: Hub, Switch, Router, Host.

2.2.3 Diseñar e implementar el acceso remoto para tener una plataforma Multiusuario, y poder compartir topologías de red por diferentes sesiones de la aplicación.

2.2.4 Implementar test virtuales, cada uno con su con su respectiva solución audiovisual

# 3. ESPECIFICACIONES

La implementación de la Capa física, Capa de Enlace de Datos, Capa de Red y Capa de Transporte tendrá las siguientes características en cuanto a las especificaciones de la lógica de negocio y de la visualización.

## 3.1 IMPLEMENTACIÓN CAPA FÍSICA.

Para la implementación de la capa física se tendrán en cuanta los siguientes aspectos de la lógica de negocio y de la visualización.

### 3.1.1 Lógica de negociación para la capa física (capa 1):

* Implementación de lógica de negocio de un cable de red tipo par trenzado UTP categoría 5, con retardos y latencias características
* Encapsulación de datos provenientes de la capa 2, en una trama de bits para que sean transmitidos al otro extremo del punto de red de la conexión.

### 3.1.2 Visualización de la Capa Física:

* Se dibujará sobre el mapa de la topología de red la conexión entre dos dispositivos, a través de la visualización de esta conexión se podrá acceder información entre ellos.
* Se mostrara un resumen de la conversión de una trama de datos que provienen de la capa de enlace de datos en bits.
* El cable de red par trenzado se encontrará ubicado, en la paleta de elementos del simulador, que permitirá la conexión entre dos dispositivos.

## 3.2 IMPLEMENTACIÓN CAPA DE ENLACE DE DATOS (CAPA 2)

Esta capa es la encargada de recibir las peticiones de la capa de Red, y utilizar los servicios de la capa física. Para la implementación de la capa de enlace de datos, se tendrán en cuenta los siguientes aspectos de la lógica de negociación y de la visualización.

### 3.2.1 Lógica de negociación para la capa de enlace de datos:

* Realizar la implementación lógica de la tramas de datos que contiene la siguiente información: MAC origen, MAC Destino, Longitud del Frame, FCS, protocolo de red en el que se originó y otros campos de control.
* Encapsulación de paquetes provenientes de la capa de red, a Tramas de capa de enlace de datos.
* Detección de Frames Corruptos, y simulación de los mismos

### 3.2.2. Visualización de la capa de enlace de datos.

* Se visualizará la encapsulación de paquetes del nivel de red a Frames, además se mostrarán todos los campos contenidos en este.

## 3.3 IMPLEMENTACIÓN CAPA DE RED (CAPA 3)

Esta capa será la encargada de la transmisión de datos en diferentes dispositivos que no estén conectados directamente, utiliza los servicios prestados por la capa 2, y le brinda servicios a la capa superior de transporte para la implementación de la capa de red se tendrán en cuenta los siguientes aspectos de la lógica de negociación y visualización.

### 3.3.1 Lógica de negociación para la capa de red

* Se realizará la implementación lógica de paquetes IP, estos deben contener información básica como: IP origen, IP destino, versión del protocolo, prioridad del paquete, longitud total del paquete, flags, tiempo de vida, el protocolo de transporte que lo origino TCP o UDP y comprobación de suma del encabezado
* Encapsulación de datagramas provenientes de la capa de transporte a paquetes IP de capa de red.
* Simulación lógica del los protocolos: IPV4, ICMP (RFC 792 ), ARP (RFC 826)

### 3.3.2 Visualización para la capa de red.

* Se visualizará la encapsulación de datagramas provenientes de la capa de transporte, en paquetes IP, también se mostraran todos los campos correspondientes del paquete.
* Visualización de eventos registrados del protocolo ICMP y de los registros del protocolo ARP.

## 3.4 IMPLEMENTACIÓN CAPA DE TRANSPORTE (CAPA 4)

Esta capa será la encargada de administrar y mantener el flujo de red entre 2 diferentes procesos encontrados en una red, para la implementación de la capa de transporte se tendrán en cuenta los siguientes aspectos de la lógica de negociación y visualización.

### 3.4.1 Lógica de negociación para la capa de transporte.

* Implementación lógica de segmentos, estos deben contener información básica como: puerto de origen, de destino, numero de la secuencia, numero de reconocimiento, longitud del encabezado, bits de código, tamaño de la ventana, Chequeo de suma, entre otros.
* Simulación del los protocolos TCP (RFC 793) y UDP (RFC 768).
* Simulación y visualización de la multiplicación de sesiones encontrada en esta capa.

### 3.4.2 Visualización de la capa de Transporte

* Visualización de segmentos con todos sus campos correspondientes.

## 3.5 IMPLEMENTACIÓN DISPOSITIVOS:

### 3.5.1 HUB

Lógica de Negociación del HUB, ubicación del dispositivo en la paleta de dispositivos de red, ubicación de uno o más HUB en el mapa de la topología de la red y visualización para acceder a la configuración del HUB.

### 3.5.2 SWITCH

Lógica de negociación, implementación de las tablas del filtrado mostrando la dirección MAC con su respectivo puerto de salida, implementación de VLANs (IEEE 802.1Q), implementación de lógica de negocio de trunking (protocolo 802.1ad), visualización del Switch, ubicación del dispositivo en la paleta de dispositivos, ubicación de uno o más switches en el mapa de la topología de la red, Visualización para acceder la configuración del Switch, como las tablas de filtrado, para la configuración de VLANs y el trunking.

### 3.5.3 ROUTER:

Lógica de negociación, implementación de tablas de enrutamiento estático, implementación protocolo RIP versión 1(RFC 1058) y versión 2 (RFC 1723), Visualización del Router, ubicación del dispositivo en la paleta de dispositivos, ubicación de uno o más Router en el mapa de la topología de la red, visualización para configurar parámetros básicos del Router, visualización para la configuración de las tablas de enrutamiento estático, visualización para configurar el protocolo RIP Versión 1 y versión 2.

### 3.5.4 HOST

Lógica de negociación para configuración básica del Host como: Dirección IP, mascara de red, puerta de enlace predeterminada, visualización del Host, visualización de la configuración básica del host.

## 3.6 ALMACENAMIENTO DE LA INFORMACIÓN DE LA TOPOLOGÍA DE RED.

* Diseño de la base de datos para almacenar diferentes topologías de red, en donde estarán ubicados los diferentes dispositivos, como sus conexiones y configuraciones.
* Lógica para la persistencia y acceso a los datos almacenados en la base de datos, este proveerá la información necesaria para instanciar los objetos de negocio que proporcionan toda la lógica de negociación de la aplicación.

## 3.7 ACCESO REMOTO DE SESIONES MÚLTIPLES DE LA APLICACIÓN.

* Diseño e implementación del acceso remoto, para que pueda ser compartida una topología de red por diferentes sesiones de la aplicación, en donde se podrán visualizar todos los dispositivos de red como sus configuraciones, por todos los usuarios que estén compartiendo la topología de la red.

# 4. MARCO CONCEPTUAL

Para delimitar el marco conceptual en el que se desarrolla este proyecto hay que hacer dos distinciones, uno es el marco teórico correspondiente a las redes de datos tipo LAN con tecnología Ethernet, el otro es el marco de referencia de programas de simulación ya existentes respecto a sistemas de comunicaciones; esto se complementará explicando brevemente nuestra plataforma de trabajo para el desarrollo del software .NET

## 4.1 REDES DE DATOS LAN TECNOLOGÍA ETHERNET

Una red datos es un sistema de intercomunicación e intercambio de información entre terminales o periféricos, las redes se originaron por la necesidad de compartir información de forma dinámica entre computadores, dando la opción que varios usuarios dispusieran de ella de forma sencilla y rápida. De esta forma el compartir información fue indispensable para el eficiente uso de las nuevas tecnologías. Hoy en día existen muchas clases de redes de datos. Estas están clasificadas según su tamaño, su método de conexión, relación funcional, tipo de transmisión, topología y según la tecnología en las que fueron implementadas que corresponde a los protocolos de intercambio de información. Entre la clasificación según el tamaño y según la tecnología tenemos:

|  |
| --- |
| * **Tipos de red según su tamaño** |
| * + Red de área personal (*PAN*) |
| * + Red de área local (*LAN*) |
| * + Red de área metropolitana (*MAN*) |
| * + Red de área amplia (*WAN*) |

Tabla N°1 Clasificación de redes según su tamaño

|  |
| --- |
| * **Tipos de red según su tecnología** |
| Ethernet IEEE 802.3 |
| Token Ring IEE 802.5 |
| Wi-Fi IEEE802.11 |
| Bluetooth IEE 802.15 |

Tabla N°1 Clasificación de redes según su tecnología

El objetivo de nuestro proyecto de grado se enfoca en la redes de Datos tipo LAN con tecnología Ethernet, usando como medio de transmisión cable de par trenzado UTP.

Un protocolo de comunicación es un conjunto de reglas y convenciones que son usadas como parámetros para establecer un canal de intercambio de información entre dos dispositivos, estos datos serán interpretados correctamente por cada dispositivo, según los protocolos que originaron la paquetes de datos y de la arquitectura del dispositivo. Esto con el fin que diferentes componentes del dispositivo entre los que se creó el canal de comunicación hablen un mismo idioma.

**Capa física**: Señal y transmisión binaria

**Capa enlace de datos**: Direccionamiento físico

**Capa de red**: Direccionamiento Lógico

**Capa Transporte**: Conexión entre dispositivos

**Capa sesión**: Comunicación entre dispositivos

**Capa de representación**: Representación de Datos

**Capa de aplicación**: Aplicaciones de usuarios

Grafica N°1 Modelo OSI

Para el estudio de este tipo de redes usaremos el modelo de referencia OSI (Open System Interconnection), este es usado como marco de referencia de la arquitectura de este tipo de redes.

Este modelo de referencia está dividido en siete capas, cada una tiene determinada el manejo de un conjunto de Protocolos que son usados para establecer un canal de comunicación entre dispositivos, estas capas se aprecian en la grafica N°1

Cada una de estas capas permite fraccionar el proceso de comunicación en una red de datos.

Dentro de las funciones y características de cada una de estas capas tenemos lo siguiente:

### 4.1.1 Capa física

La capa física es la encargada de las conexiones físicas entre los equipos de la red. En esta se definen las características y los valores nominales (como voltajes, polaridad, tiempo de duración del bit y otras) del medio por el que se transfiriere la información.

### Capa de enlace

Esta es la capa encargada de recibir los bits provenientes de la capa física y comprobar si durante la transmisión del mensaje hubo alguna variación de este. Esta capa fragmenta los mensajes que se están recibiendo, los analiza y determina si hay algún error en el mensaje recibido, por medio de protocolos con algoritmos que gestionan dichos bits erróneos, perdidos o duplicados. En esta capa también se maneja la dirección física de los dispositivos, esta sirve cómo identificación para cada puerto Ethernet, este número es único e irrepetible por cada uno.

### 4.1.3 Capa de red

Esta capa determina cual es el mejor camino que debe usarse para enrutar los paquetes transmitidos, aquí se incorpora la dirección lógica, origen y destino de la información. Existen diferentes algoritmos tanto dinámicos como estáticos, que determinan cual es la mejor ruta, dependiendo de diferentes variables como son tiempo, longitud del camino y estado del mismo.

### 4.1.4 Capa de transporte

Esta es la primera capa donde hay comunicación directa entre host. El objetivo es transportar eficientemente la información. Aquí se identifica cual fue el proceso o programa para lo que fue enviada la información. Esto es posible, ya que toda aplicación que se ejecuta en el host transfiere su información a través de un puerto identificado y distinguido de forma única, así si el host está ejecutando varias aplicaciones la capa de red sabe a cual pertenece cada paquete de información recibido y enviado.

### 4.1.5 Capa de sesión

De forma complementaria a la capa de transporte, la capa de sesión maneja servicios adicionales para la mejor gestión de la transmisión de la información, Por ejemplo, puede manejar “tokens (objetos abstractos y únicos) para controlar las acciones de participantes o puede hacer checkpoints (puntos de recuerdo) en las transferencias de datos”.

### 4.1.6 Capa de presentación

En esta capa se gestionan protocolos y funciones comunes para muchas aplicaciones tales como traducciones entre caracteres, códigos de números, etc.

### 4.1.7 Capa de aplicación

Finalmente en esta capa son gestionados los protocolos usados por las aplicaciones individuales.

Estos son los aspectos más relevantes que debemos tener en cuenta a la hora de comenzar nuestra investigación para el desarrollo de nuestro simulador. Existe mucha información referente a este tema, en esta se analiza más detalladamente cada una de las características de los protocolos de red existentes. Para nuestro objetivo es importante dar a conocer los aspectos relevantes desde nuestro punto de partida para el desarrollo de nuestro software.

## 4.2 TEORÍA DE OTROS SIMULADORES

## Software actual para la simulación de redes de comunicaciones

Para ubicar puntualmente el software de simulación de nuestro proyecto hicimos un estudio detallado de algunas de las principales y más importantes herramientas de software para la simulación de redes de comunicaciones, estas son:

* FLAN es un software de propósito general para la simulación de redes de comunicación, este programa hace el análisis de las redes asociando su estructura basada en nodos y enlaces, con bloques simples, por medio de los cuales se puede entender el funcionamiento especialmente de los protocolos de enrutamiento que maneja la capa de red. La limitación es que el usuario debe contar con conocimientos básicos de programación en Java, en el momento de definir características y parámetros de los dispositivos.
* PACKET TRACER es un simulador gráfico de redes desarrollado y utilizado por Cisco como herramienta de entrenamiento para obtener la certificación CCNA (Cisco Certified Network Associate), este ofrece como ventaja principal el análisis de la simulación de acuerdo al modelo OSI de capas, pero tiene la limitación que es un software de tipo propietario y habría que pagar la licencia para su utilización además que su uso es exclusivo para dicho entrenamiento. Además no permite crear topologías de red que involucren la implementación de tecnologías diferentes a Ethernet.
* OPNET MODELER este paquete de simulación es muy utilizado en la industria del diseño de redes de comunicaciones, muy utilizado en la investigación y desarrollo, es un software de tipo propietario, diseñado especialmente para la industria por lo que el tiempo para el aprendizaje y manejo de este software es largo y riguroso.
* KIVA es un simulador de redes basado en Java que permite especificar diferentes esquemas de redes de datos y simular el encaminamiento de paquetes a través de dichas redes, esta es una herramienta software orientada principalmente a simular el comportamiento del protocolo IP, y especialmente para el estudio del tratamiento de los datagramas y el encaminamiento de los mismos por una red. En la versión actual, la interfaz de usuario está implementada con un conjunto de clases, las cuales deben ejecutarse en el equipo del usuario, además, cada vez que se desee trabajar con éste programa se deben descargar varios archivos para poder instalar el programa.
* COMNET III es una herramienta comercial orientada al diseño, configuración y estudio de las redes de comunicaciones, desarrollado por CACI Products Inc. Por medio de este programa es posible crear topologías de redes complejas, configurar varias tecnologías, protocolos y dispositivos de red, para hacer un análisis detallado del funcionamiento y del rendimiento de redes tipo LAN, MAN y WAN. Como desventaja principal tenemos que es un software propietario. Por ser una de las herramientas de simulación más completas del mercado, la programación de los parámetros de los dispositivos y enlaces de la red tiende a ser compleja. Además de los conocimientos sobre el manejo y el diseño de redes de comunicaciones, se requieren conocimientos en otras áreas como por ejemplo la estadística.

Con el estudio de las características y componentes del software anteriormente expuestos y las necesidades particulares de la materia de integración de redes, definimos y delimitamos las características de software del simulador que desarrollaremos en nuestro proyecto. La ventaja más grande que tiene nuestro proyecto en comparación con los simuladores actuales, es el diseño y la plataforma sobre la que se va a diseñar el software, este se implementara sobre la plataforma .NET, diseñada por Microsoft este ambiente de diseño se escogió por contar con un “Entorno Común de Ejecución para Lenguajes” o CLR (Common Language Runtime), tal como se describe en breve.

## 4.3 .NET FRAMEWORK Y EL ENTORNO COMÚN DE EJECUCIÓN PARA LENGUAJES (CLR)

Uno de los ideales de la ciencia de la computación a través de los últimos años, ha sido el concepto de que un programa de computadora puede ser escrito para que sea ejecutado sin importar el Hardware con el que este compuesto.

Hasta cierto punto este ideal se ha logrado al poder abstraer el acceso al Hardware utilizando un sistema operativo, desde este punto de vista la función básica de un sistema operativo es encapsular la capa de Hardware de un sistema, para que un programador no tenga la necesidad de preocuparse por registros, memoria, y otros problemas que ocurren a nivel de Hardware.

En los últimos años, este problema volvió a surgir, pero esta vez ya no a nivel de Hardware, sino a nivel de un sistema operativo, en el mercado actual se encuentran diferentes sistemas operativos Win32, Solaris, OSs, inclusive de un mismo Proveedor como por ejemplo Microsoft contiene una serie de sistemas operativos diferentes (Win98, WinXp, WinVista). El problema radica ahora en darle al programador, otro nivel de abstracción en donde esté encapsulado el sistema operativo cuando se está desarrollando una aplicación. En muchos sentidos esto es exactamente lo que realiza Microsoft con .NET.

El Entorno Común de Ejecución para Lenguajes o CLR (Common Language Runtime) por sus siglas en ingles, el cual es uno de los pilares del Framework o marco de trabajo de .Net , forma un ambiente que abstrae el sistema operativo.

MARCO DE TRABAJO .NET

SISTEMA OPERATIVO

SISTEMA OPERATIVO

HADWARE

HADWARE

HADWARE

Gráfica N° 2 ambiente de abstracción del sistema operativo.

.NET abstrae los sistemas operativos, los cuales abstraen el acceso a Hardware, esto significa que al desarrollar software, se está escribiendo código encaminado al CLR y no directamente al sistema operativo, por lo que este software puede correr sobre diferentes sistemas operativos o cualquier plataforma que implemente el CLR. El CLR está presente en el núcleo del Framework de .NET, este provee un ambiente en donde las aplicaciones son ejecutadas, esto incluye conceptos como compilación, registro y hasta problemas de desarrollo.

Como su nombre lo implica el CLR está diseñado para soportar diferentes lenguajes de programación de forma en común entre ellos. Esto es muy interesante ya que los desarrolladores de software no tienen que aprender un nuevo lenguaje de programación, para trabajar sobre .NET.

Microsoft creó los siguientes lenguajes de programación encamidos para trabajar sobre .NET, los principales son: VB,C#,C++, JScript. Otras firmas están trabajando para desarrollar lenguajes de programación encaminados a .NET como COBOL, Phyton, y posiblemente JAVA.

# 5. DESCRIPCIÓN GENERAL Y DIAGRAMA EN BLOQUES

Nuestro proyecto simulará los aspectos más importante que ocurren en una red de datos tipo LAN con tecnología Ethernet, para el desarrollo de ese simulador vamos a utilizar la plataforma de desarrollo .NET ya que esta herramienta cuenta con todos los aspectos tecnológicos y necesarios para alcanzar los objetivos de este proyecto. Unos de los aspectos mas relevantes en la escogencia de este software, es su capacidad y flexibilidad para futuras mejoras de nuestro proyecto, por otra parte posee las herramientas necesarias para elaborar una plataforma multiusuario y de esta forma cumplir con nuestro objetivo de ser un sistema multiusuario, para ser usado de forma didáctica siendo un sistema de ámbito académico.

Para el desarrollo del software utilizaremos la siguiente metodología:

## 5.1 DESARROLLO DE LA APLICACIÓN EN CAPAS:

Una característica de nuestro software es que sea extensible sencillamente, ya que servirá como base para implementaciones especificas, o para la actualización de nuevas tecnologías. Cuando sea necesario implementar un nuevo módulo, no se tendrá que volver a escribir todo el código; el diseño de esta aplicación debe permitir la reutilización del código, y que sólo sea necesario codificar nuevas implementaciones, permitiendo que la aplicación sea extensible de una forma rápida para futuros proyectos que requieran ser complementados.

Se podría hacer funcionar la aplicación en forma rápida, haciendo caso omiso de ningún patrón de desarrollo de software por capas, dejando todos los módulos que la compongan altamente acoplados entre ellos, esto conllevaría a que sería imposible la extensión de esta aplicación, o que fuera necesaria volver a escribir gran parte de código, por lo que se podria correr el riesgo que la aplicación quede limitada a las funcionalidades que se van a realizar con este proyecto.

Otro inconveniente que se encuentra al no utilizar el diseño por capas, sería cuando la tecnología de .NET este obsoleta, o sean creados o actualizados nuevos módulos, sería imposible estar a la altura de estos cambios, esto conllevaría a lo que ocurre con muchas aplicaciones, que después de un tiempo serían obsoletas porque no pueden ser utilizadas debido a un cambio de tecnología( como el cambio de Windows XP a Windows Vista), o si se desea utilizar nuevas tecnologías es imposible acondicionarlas, debido a que todos los componentes de la aplicación están altamente acoplados, y un cambio en cualquiera de ellas involucra un cambio en todos los componentes de la aplicación.

El otro inconveniente que encontramos al no diseñar por capas, es la dificultad para realizar pruebas unitarias a los módulos que componen la aplicación, porque estos, entre si están altamente acoplados.

Por estas razones se usará un modelo en capas en donde esté altamente diferenciado, la visualización, el modelo de negociación, y el acceso a los datos.

La visualización es la capa encargada de la interfaz con el usuario, y su única responsabilidad será la de recibir cualquier evento que sea originado por el usuario, o desplegar la información necesaria al usuario. En este caso se utilizará para la capa de visualización, ventanas de Windows para la representación de la información.

La capa de negociación es la encargada del manejo de cálculos y procesos de los datos, esta capa no conoce la existencia de una visualización, o una capa de acceso a datos, su única función es la de realizar toda la lógica de negociación correspondiente. Esta capa será el núcleo de la aplicación en donde verdaderamente estará ocurriendo toda la lógica de la simulación.

En la capa de acceso a datos, está la abstracción de la fuente de datos, que va hacer usada por la capa de Negociación, esta capa obtendrá los datos de la topología de la red, desde un archivo, o una base datos o usando .NET Remoting, para obtener los datos de red de la capa de datos de una aplicación remota del simulador.

Con esta división por capas, es sencilla la extensión de la aplicación debido a que están altamente desacoplados los módulos principales, de esta manera si en un futuro se requiere presentar este simulador en un ambiente WEB, lo único necesario será implementar la capa de visualización en un entorno WEB, la capa de negociación y el acceso a datos quedarán intactas.

Si en un futuro la información que maneja el simulador es demasiado grande, en donde el modelo de acceso a datos de la configuración de la red, a través de archivos no pueda ser mantenida, y sea necesaria la utilización de una base de datos relacional, tan solo sería necesario cambiar la capa de acceso a datos, para que soportara este nuevo formato, las restantes capas quedan intactas, y la aplicación continuará funcionando normalmente.

Lo mismo ocurre al nivel de la capa de negociación, cuando se requiera hacer la implementación de un nuevo protocolo, o cambiar la lógica de algún dispositivo, esto no debe afectar las demás capas.

## 5.2 COMUNICACIÓN ENTRE LAS DIFERENTES CAPAS

Para la comunicación entre las 3 capas se va a usar la metodología Modelo Vista Presentación. Esta metodología, es muy recomendable para desacoplar las capas de visualización de negociación, como también para que se puedan realizar pruebas unitarias tanto a la visualización como al modelo de negociación.

El modelo es el objeto donde residen los datos de la aplicación y provee métodos para acceder a ellos, en este también esta toda la lógica de negociación, el modelo no debe tener ninguna asociación con la interfaz con el usuario, no contiene una referencia a la vista, el modelo envía eventos cuando sus datos cambian, en el caso del simulador, van a existir diferentes modelos, cada uno para los diferentes dispositivos que existan y diferentes protocolos a implementar.

El nivel de vista es el encargado de mostrar los datos del modelo al usuario, en el caso del simulador la vista será una ventana de Windows, en donde estará el mapa de la topología de red, y la información de cada dispositivo, la vista notificará mediante eventos, cuando intervino el usuario.

La presentación es la encargada de sincronizar y es la mediadora entre la vista y el modelo, la presentación conoce tanto al modelo como a la visualización y continuamente está escuchando los eventos de acción del usuario; una vez la presentación detecte este evento leerá los datos de la visualización y los transportará al modelo, la presentación también escucha los eventos de cambio del modelo, para que cuando ocurran, la visualización sea refrescada con los cambios que ocurrieron en el modelo.

En la aplicación se usará esta metodología por cada componente que exista, por lo que tendrán varios MVPs, y existirá un modelo global MVP, el cual contendrá varios MVPs , que en este caso será el mapa de la red.

Grafica N° 2

Para la implementación remota entre diferentes sesiones del simulador se usara .NET remoting junto con el modelo MVP, en donde una sesión del simulador se convertirá en el servidor, y en este estará todo el modelo de red de todas las sesiones las cuales serán los clientes, en el servidor también se encontrará el presentador, y una vista, cada uno las demás sesiones que estén conectadas entre sí, solo serán visibles.

Presentación

Modelo

Vista

Host

Host

Vista

Presentación

Modelo

TCP

TCP

HTTP

HTTP

Grafica N° 3

Mediante este modelo estableceremos la conexión entre diferentes sesiones del simulador, en donde será posible la comunicación entre diferentes componentes de red de las sesiones que estarán contenidas en un mismo modelo de red la cual se encontrará en el servidor.

## 5.3 DIAGRAMA EN BLOQUES:

En el siguiente diagrama se sintetiza todos los aspectos más relevantes del presente proyecto de grado.

Grafico 3. Diagrama en bloques del proyecto de grado.

# 6. DESARROLLO TEÓRICO

En el desarrollo y diseño del software de simulación nos basaremos en técnicas y metodologías utilizadas para la planeación y estructuración de un plan de trabajo con el fin de hacer del proceso de programar un proceso ágil y eficiente.

Para el desarrollo de esta aplicación, se van a utilizar metodologías de desarrollo ágil de software, principalmente XP (eXtreme Programming) y SCRUM, estas metodologías han comprobado ser la mejor opción para el desarrollo de software al compararlos con metodologías tradicionales de desarrollo de software.

En las metodologías tradicionales existen diferentes fases bien definidas, que consisten en la recopilación de los requerimientos de la aplicación, análisis de los requerimientos, diseño de una solución, diseño de la aplicación, desarrollo de la codificación, pruebas, puesta en producción, y post producción.

El problema de utilizar esta metodología, radica en que las fases solo ocurren una vez durante todo el tiempo del desarrollo del proyecto, esto conlleva a que los requerimientos del producto, son recopilados en un solo instante, en donde ambas partes tanto el cliente como el equipo de desarrollo, no tienen bien identificado el problema, o tienen una idea vaga de lo que el producto debe realizar; una vez culminada esta etapa de recopilación se procede con el diseño de la aplicación, hasta que el diseño no esté terminado no se procede a la codificación, y por último se realizan las pruebas. Lo más probable es que al final del proyecto, el cliente no esté satisfecho, con la producción final del proyecto, aunque verdaderamente el producto cumpla con los requerimientos que fueron recopilados en la etapa inicial, debido a que no existe una constante entrega del funcionalidades parciales del producto al cliente, sino que hace la entrega total del proyecto al final del desarrollo de la aplicación.

Básicamente las metodologías agiles de software, como XP o Scrum, proveen una solución de este problema, haciendo ciclos más rápidos de producción, y procurando proporcionar un contacto más frecuente con el cliente, en donde constantemente se estén entregando funcionalidades del producto, para que el cliente pueda dar su opinión, acerca del producto, Al realizar esto se habilitará la opción de realizar realimentaciones a través del tiempo, el cliente podrá descubrir verdaderamente lo que necesita que el producto realice, por lo que al final del proyecto el cliente estará satisfecho con el producto.

## 6.3 CONTROL DE VERSIONES:

El control de versiones es el mecanismo para gestionar las diferentes versiones de todos los elementos de configuración que forman la línea base de un producto o la configuración del mismo.

El control de versiones provee el mecanismo de almacenamiento de cada uno de los temas que desea gestionarse. Este habilita la posibilidad de modificar, mover, borrar cada uno de los elementos. Mantener un historial de todas las acciones realizadas con cada elemento pudiendo volver a un estado anterior dentro de este historial.

Mediante este control se tiene la posibilidad de retornar la producción del software a un estado anterior, esto puede ser necesario cuando en algún momento del desarrollo, se detecte un error en el diseño y sea necesario volver a una revisión anterior en donde se originó el error. Este control de versión es muy útil, en el desarrollo de software, ya que sirve como documentación del código, y brinda la posibilidad de volver a una versión anterior y analizar lo ocurrido.

Cada vez que se genera una versión existe la posibilidad de proveer detalles acerca de los cambios ocurridos en esta versión, de esta forma quedará documentado todo cambio que ocurra en la aplicación. Todo este historial y todos los archivos y elementos involucrados se encuentran en un repositorio, que se encuentra ubicado en un servidor, y puede ser accedido en cualquier parte a través de internet.

Esto posibilita las futuras complementaciones del producto, los nuevos desarrolladores puedan tener acceso a este repositorio y tengan un conocimiento mas profundo de cómo fue creado el producto desde su fase inicial, como información de su evolución hasta su fase final, y puedan seguir trabajando sobre este repositorio La herramienta que utilizaremos para el control de versionamiento, es SubVersion, el cual es de libre distribución.

# 7. PRUEBAS DE CALIDAD Y DESEMPEÑO

Para tener control de la calidad de nuestro diseño de software, y tener la total seguridad que cada una de las funciones de nuestro simulador funcione a la perfección usaremos el desarrollo de software encaminado a pruebas, TDD (Test Driven Development) de la metodología de desarrollo XP, en esta metodología cada nuevo módulo que se desee implementar comienza con la escritura de pruebas del modulo, las cuales obligatoriamente deben fallar debido a que no ha sido escrito ningún código para la funcionalidad de este. Las pruebas tienen que estar muy bien diseñadas, para que cubran todas las funcionalidades que este modulo exponga.

El siguiente paso, es escribir el código para hacer pasar las pruebas, sin importar la forma en que se escriba el código, aunque no sea de una forma perfecta, se debe escribir únicamente para hacer pasar las pruebas.

Consecutivamente se vuelven a correr las pruebas a satisfacción. A continuación se reestructurara el código: Una vez son pasadas las pruebas, se procede a reestructurar el código para que no se repitan parte de este, se aplican patrones de programación, para que sea un código entendible y extensible, en esta etapa existe la seguridad para cambiar el código de los módulos, sin que exista el temor de dañar alguna funcionalidad, ya que las pruebas realizadas proveerán la información necesaria para saber si el módulo está funcionando correctamente.

Este ciclo se repite constantemente en todas las etapas del proyecto, al final de este existirán las pruebas unitarias de todos los módulos.

Para futuros desarrollos de la aplicación que requieran cambios en el núcleo principal, no existirá la preocupación de dañar algún modulo que esté funcionando correctamente sin darse por enterado, ya que las pruebas unitarias fallaran en el sitio en donde ocurrió el error. Como parte final de nuestro proyecto hemos querido diseñar unos test virtuales con soluciones audiovisuales, estos con el fin de cubrir cada una de las funciones de nuestro software. Con esta herramienta cubrimos dos objetivos del simulador. Tener el manual de usuario en donde se explique cada una de las funciones del simulador, y la otra es ver estas funciones en funcionamiento y puestos en práctica.

# 8. COSTOS Y FUENTES DE FINANCIACIÓN

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **COSTOS UNIVERSITARIOS** | | | | | |
| Recursos Humanos | **Unidad** | **Cantidad** | **Valor Unitario** | **Valor Total** | **Fuente** |
| Director del Proyecto | Horas (Semana) | 1 | $ 75.000 | $ 1.800.000 | Pontificia Universidad Javeriana |
| Asesor de Tesis | Horas (Semana) | 1 | $ 75.000 | $ 1.800.000 | Pontificia Universidad Javeriana |
| Alquiler Computador | Mensual | 0 | $ 600.000 | $ 0 | Pontificia Universidad Javeriana |
| Internet | Horas | 600 | $ 500 | $ 300.000 | Pontificia Universidad Javeriana |
| Libros | Indefinido | 6 | $ 200.000 | $ 1.200.000 | Pontificia Universidad Javeriana |
| **Valor Total Costos Universitarios** | | | | **$ 5.100.000** |  |
|  |  |  |  |  |  |
| **COSTOS INDIVIDUALES** | | | | | |
| Recursos Humanos | **Unidad** | **Cantidad** | **Valor Unitario** | **Valor Total** | **Fuente** |
| Aspirante | Horas (Semana) | 42 | $ 25.000 | $ 25.200.000 | Recursos propios |
| **Valor Total Costos Individuales** | | | | **$ 25.200.000** |  |
|  |  |  |  |  |  |
| **COSTOS TOTALES** | | | | **$ 30.300.000** |  |

# 9. JUSTIFICACIÓN DE POSIBLES DIFERENCIAS ENTRE LA PROPUESTA Y EL PROYECTO

Para el diseño de un software se ha mantenido tradicionalmente en la industria, una gran diferencia entre el producto esperado y el producto entregado, esto es dado a que las tendencias tradicionales de programación funcionan de forma estática sin realimentación del usuario directo en el momento de diseñar. De esta forma el usuario al tener una necesidad le daba la tarea al programador que diseñará un programa para satisfacer dicha necesidad. Pero durante el tiempo en que el programador demora en elaborar su nuevo software, este usuario se ha encontrado con muchas más necesidades que hubieran podido ser cubiertas y solucionadas de una vez con el programa que se está desarrollando, de esta forma para la entrega final del software, efectivamente el requerimiento inicial fue satisfecho pero en la brecha de tiempo las necesidades que han surgido hacen que el producto ya este por debajo de las necesidades actuales.

Por esta razón nosotros en el desarrollo del software propuesto en este trabajo de grado hemos querido que estas diferencias sean beneficiosas para todos las personas que estas involucradas en el proyecto. Nuestro principal lineamiento en el desarrollo del software es la satisfacción del cliente, en este caso nuestro director de Proyecto de grado y los estudiantes de la carrera de Ingeniería Electrónica. El uso de herramientas ágiles para el desarrollo de software XP y SCRUM, implican una comunicación directa y continua con el usuario del programa y entregas individuales de forma continua, de esta manera si en el momento de cumplir con uno de los requerimientos iniciales surge la necesidad de cubrir otra necesidad que no se hubiera planteado de forma simultánea se puede resolver, de esta forma ganamos tiempo en el diseño y el producto final es el más satisfactorio y funcional para las necesidades del momento.

# 10. BIBLIOGRAFIA Y FUENTES DE INFORMACIÓN

[1] Rockford Lhotka: Expert C# 2005

Business Objects, Second Edition, Apress 2006

[2] Joe Duffy: Professional .NET Framework 2.0. First Edition, Wrox Press 2006

[3] Jeffrey Richter: Applied Microsoft .NET Framework Programming, First Edition Microsoft Press 2002

[4] Ingo Rammer: Advanced .NET Remoting (C# Edition), First Edition, Rammer Apress © 2002

[5] Andy Hunt: Pragmatic Unit Testing in C# with NUnit, First Edition, The Pragmatic Bookshelf 2003

[6] Todd Lamle: CCNA: Cisco Certified Network Associate Study Guide, 5th Edition, Sybex 2005

[7]. Modelo MVP, obtenido 22 octubre en:

<http://www.codeproject.com/aspnet/ModelViewPresenter.asp>

[8]. [Douglas E. Comer](http://www.amazon.com/exec/obidos/search-handle-url/002-9655500-1632011?%5Fencoding=UTF8&search-type=ss&index=books&field-author=Douglas%20E.%20Comer): Internetworking with TCP/IP Vol.1: Principles, Protocols, and Architecture (4th Edition), Prentice Hall 2000

[9]. Imagen NUnit, obtenido 22 octubre en:

<http://www.codeproject.com/dotnet/tdd_in_dotnet.asp>

# 11. OBSERVACIONES

Es importante anotar que la visión del desarrollo de nuestro proyecto no finaliza con la realización y entrega de mismo por parte de nosotros, hemos querido que nuestro trabajo de grado sea un inicio de una línea de desarrollo e investigación. En un futuro al llegar nuevas tecnologías y tendencias de redes de comunicaciones nuestro simulador no quedará obsoleto y en el olvido, el diseño del código y la programación por capas hace que el código pueda ser extensible y mejorado fácilmente, simplemente con la adición de las nuevas funciones, o con la modificación de alguna ya desarrollada, tenemos total seguridad que no va a influir negativamente en cualquier otra función del programa, debido a que cada función particular del software esta encapsulada y es transparente lo que pasa en su interior a las otras funciones que comparte en software en común.

Para el desarrollo del diseño del software utilizaremos un repositorio en la internet, de forma que todo el historial de las líneas de código y de avances y correcciones que tenga el código del programa puedan ser consultadas por cualquier persona que este interesado en este. De esta forma los futuros estudiantes que estén interesados en este desarrollo sea muy sencillo enterarse de todo el desarrollo que se tuvo en el desarrollo de este programa. Es de anotar que en el uso de la plataforma .NET los futuros usuarios podrán diseñar los nuevos complementos en el lenguaje de programación que mejor tengan dominio, ya que esta plataforma de desarrollo tiene la capacidad de manipular varios leguajes de programación teniendo el mismo producto final.

# 12. ANEXOS

Imágenes del producto final