UNIVERSIDAD DE SONORA

CARRERA:

INGENIERA EN SISTEMAS DE INFORMACION

REPORTE FINAL DE PRACTICAS PROFESIONALES

ALUMNO:

MANUEL CELESTINO AGUILAR OSUNA

EXPEDIENTE:

208203119

PRACTICAS REALIZADAS EN:

CENTRO DE SERVICIOS Y TECNOLOGIAS DE INFORMACION

PROYECTO:

ADMINISTRACION DEL EQUIPO DE CÓMPUTO DEL LABORATORIO CENTRAL DE INFORMATICA DE LA UNIVERSIDAD DE SONORA.

miércoles, 06 de marzo de 2013

Índice.

Tabla de ilustraciones

1. INTRODUCCIÓN

El presente proyecto se refiere al desarrollo de aplicaciones para el bloqueo del equipo de cómputo de laboratorio central de informática de la universidad de sonora para la correcta utilización de los equipos de dicho laboratorio.

En este documento se mencionan aspectos importantes que fueron tomados en cuenta para la elaboración de la investigación, así como también, las decisiones que fueron tomados para cada una de las alternativas a la problemática.

También se explica cómo fue desarrollado el proyecto describiendo las herramientas y aplicaciones utilizadas para su elaboración.

1.1 DESCRIPCIÓN DEL ÁREA DE LA INSTITUCIÓN EN LA QUE DESARROLLÓ LA PRÁCTICA

Para la elaboración de mis prácticas profesionales se me asignó al área de cliente-servidor del proyecto para la administración de equipo de cómputo del laboratorio central de informática.

En esta área, fui encargado primeramente hacer una investigación sobre las aplicaciones cliente servidor su estructura y ambiente en .NET Visual C#.

Esta área es la encargada es la encargada de limitar el acceso a los equipo de cómputo bloqueándolos de las funciones.

El laboratorio central de informática (LCI) es el área donde se proveen o brindan servicios como el uso del equipo de cómputo para uso escolar, registro de correo electrónico, servicios complementarios y conectividad a Internet a los alumnos de la universidad de sonora y también servicios de impresión.

1.2 JUSTIFICACIÓN DEL PROYECTO REALIZADO

Dentro de laboratorio central de informática de la universidad de sonora se cuenta con los equipos de cómputo para brindar el servicio de uso escolar a los alumnos de la universidad de sonora.

Los principales usos de los usuarios de este laboratorio son:

a) Utilización de diferentes tipos de software vinculado a su correspondiente carrera.
b) Uso de navegadores de internet para investigaciones.
c) Tareas escolares.
d) Revisar cuentas de correo institucional

Este proyecto consta en la realización del desarrollo e implementación de una aplicación cliente y una aplicación servidor para mejorar la gestión y administración de los equipos de cómputo y restringir acceso a dichos equipos por medio del bloqueo.

Los principales procesos que se llevan a cabo dentro del laboratorio central de informática son los siguientes:

a) Préstamo de equipos a los usuarios.
b) Administración del tiempo de los usuarios (3 horas diarias)
c) Asignación manual por parte de los encargados a los equipos de cómputo.
d) Autoacceso a los usuarios a través del lector de credenciales.

Con las aplicaciones cliente y servidor se mejorará la gestión y administración tanto de acceso a los computadoras del LCI como también se llevará un mejor control del tiempo de utilización de los equipos y el software más utilizado por los usuarios además de que se difundirá una cultura de credencialización ya que es desbloqueo de la computadora será a través del número de expediente. Por lo tanto se mejorara también la atención en el servicio de soporte técnico y la contabilidad para los presupuestos para más equipos y software.

1.3 OBJETIVOS DEL PROYECTO.

En general el objetivo es bloquear el equipo de cómputo del laboratorio central de informática para que los usuarios se vean a obligados tanto a resellar su credencial como también memorizar su expediente institucional para poder hacer uso de los equipos.

Para que el objetivo del proyecto se cumpla se requiere que:

a) Mejorar la administración y gestión del acceso a los equipos de cómputo del LCI.
b) Mejorar el servicio técnico a los usuarios del LCI.
c) Mantener actualizada la información sobre el uso del tiempo.
d) Bloquear a través
e) Difundir una cultura de resello de credencial y uso del expediente institucional.

1.4 PROBLEMAS PLANTEADOS A RESOLVER

En el laboratorio central de informática se cuenta con 261 equipos de cómputo para uso escolar repartidos en diferentes salas en mesas con diferente número de equipos por mesa. El problema principal reto que se me presentó fue la comunicación dentro de la red de los equipos, tanto para las maquinas cliente como para las computadoras que serían Servidores que bloquearán a las primeras. Esto implicación la apertura y comunicación por puertos en ambas partes.

En base a lo anterior, los principales problemas son:

a) Restricción del acceso a los equipos de cómputo en LCI.
b) Falta de cultura por el resellado de la credencial institucional.
c) Tiempo excedido del uso de los equipos de cómputo.
d) Todos los usuarios tienen acceso a todas los equipos de cómputo sin restricción de tiempo.
e) La totalidad de los equipos de cómputo tienen software con licencias limitadas que deben ser gestionadas.

1.5 Alcances y limitaciones en la solución de problemas.

1.5.1 Alcance:

Brindar un mejor servicio para restringiendo el acceso a los equipos de cómputo por usuarios no autorizados.

1.5.2 Limitaciones:

- Las aplicaciones tanto cliente como servidor no cuentan con la base de datos del LCI por lo que su funcionamiento no se asocia por el momento con los expedientes.
- Sistemas operativos diferentes o no compatibles con las aplicaciones desarrolladas.
- Fallas en la conectividad en la red por el protocolo TCP/IP.

2. FUNDAMENTO TEÓRICO DE LAS HERRAMIENTAS Y CONOCIMIENTOS APLICADOS.

A continuación se describen técnicas y herramientas utilizadas para facilitar las actividades que se me asignaron durante mis prácticas profesionales, para comprender mejor se describe brevemente cada herramienta utilizada y las actividades llevadas a cabo.

2.1 Lenguaje de programación utilizado para el proyecto C Sharp

C Sharp es el nuevo lenguaje de propósito general orientado a objetos creado por Microsoft para su nueva plataforma .NET. C Sharp combina los mejores elementos de múltiples lenguajes de amplia difusión como C++, Java, Visual Basic o Delphi. De hecho, su creador Anders Heljsberg fue también el creador de muchos otros lenguajes y entornos como Turbo Pascal, Delphi o Visual J++. La idea principal detrás del lenguaje es combinar la potencia de lenguajes como C++ con la sencillez de lenguajes como Visual Basic, y que además la migración a este lenguaje por los programadores de C/C++/Java sea lo más inmediata posible.

Además de C Sharp, Microsoft proporciona Visual Studio.NET, la nueva versión de su entorno de desarrollo adaptada a la plataforma .NET y que ofrece una interfaz común para trabajar de manera cómoda y visual con cualquiera de los lenguajes de la plataforma .NET (por defecto, C++, C Sharp, Visual Basic.NET y JScript.NET, aunque pueden añadirse nuevos lenguajes mediante los plugins que proporcionen sus fabricantes).

2.2 Conectividad de red utilizando el comado *ping*.

La herramienta Ping "Ping" (forma abreviada de Packet Internet Groper) es sin duda la herramienta de administración de redes más conocida. Es una de las herramientas más simples ya que todo lo que hace es enviar paquetes para verificar si una máquina remota está respondiendo y, por ende, si es accesible a través de la red.

La herramienta ping permite de esta manera diagnosticar la conectividad a la red mediante comandos del tipo:

ping nombre.del.equipo

name.of.the.machine representa la dirección IP de la máquina, o su nombre. Por lo general, se recomienda hacer una prueba usando la dirección IP de la máquina en primer lugar.

2.3 Funcionamiento de ping

Ping depende del protocolo ICMP, el cual permite diagnosticar las condiciones de transmisión. Utiliza dos tipos de mensajes de protocolo (de los 18 que ofrece ICMP):

El tipo 0, corresponde a un comando "solicitud de eco" enviado por la máquina fuente.

El tipo 8, corresponde a un comando "solicitud de eco" enviado por la máquina destino.

Con intervalos regulares (predeterminados por segundo), la máquina fuente (la que ejecuta el comando ping) envía una "solicitud de eco" a la máquina destino. Cuando se recibe el paquete "respuesta de eco", la máquina fuente muestra una línea que contiene cierta información. En caso de no recibir una respuesta, aparecerá una línea indicando que "el tiempo de espera de la solicitud ha finalizado".

Por lo tanto, la salida del comando ping permite conocer:

- La dirección IP que corresponde al nombre de la máquina remota.
- El número de secuencia ICMP.
- La vida útil del paquete (TTL). El campo de vida útil (TTL) permite conocer la cantidad de routers por los que pasó el paquete mientras viajó de una máquina a otra. Cada paquete IP posee un campo TTL con un valor relativamente alto. Cada vez que pasa por un router, se reduce

el valor. Si alguna vez este número es cero, el router interpretará que el paquete está viajando en círculos, por lo tanto, finaliza el proceso.

- El campo de demora de vueltas corresponde al lapso de tiempo en milisegundos que se necesita para dar una vuelta entre las máquinas fuente y destino. Cómo regla general, la demora de un paquete no debe ser mayor a 200 ms;
- La cantidad de paquetes perdidos.

2.4 CLIENTE SERVIDOR

Desde el punto de vista funcional, se puede definir la computación Cliente/Servidor como una arquitectura distribuida que permite a los usuarios finales obtener acceso a la información en forma transparente aún en entornos multiplataforma. En el modelo cliente servidor, el cliente envía un mensaje solicitando un determinado servicio a un servidor (hace una petición), y este envía uno o varios mensajes con la respuesta (provee el servicio).En un sistema distribuido cada máquina puede cumplir el rol de servidor para algunas tareas y el rol de cliente para otras.

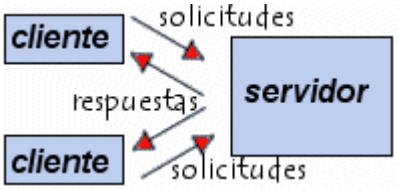


Figura 2.1 Modelo Cliente Servidor

En otras palabras la arquitectura Cliente/Servidor es una extensión de programación modular en la que la base fundamental es separar una gran pieza de software en módulos con el fin de hacer más fácil el desarrollo y mejorar su mantenimiento.

2.4.1 CLIENTE

El cliente es el proceso que permite al usuario formular los requerimientos y pasarlos al servidor, se le conoce con el término front-[1]end . El Cliente normalmente maneja todas las funciones relacionadas con la manipulación y

[1] es la parte de un sistema de software que interactúa directamente con el usuario

despliegue de datos, por lo que están desarrollados sobre plataformas que permiten construir interfaces gráficas de usuario (GUI), además de acceder a los servicios distribuidos en cualquier parte de una red.

Las funciones que lleva a cabo el proceso cliente se resumen en los siguientes puntos:

• Administrar la interfaz de usuario.

• Interactuar con el usuario.

• Procesar la lógica de la aplicación y hacer validaciones locales.

• Generar requerimientos de bases de datos.

• Recibir resultados del servidor.

• Formatear resultados

2.4.2 SERVIDOR

Es el proceso encargado de atender a múltiples clientes que hacen peticiones de algún recurso administrado por él. Al proceso servidor se le conoce con el término back-end[2]. El servidor normalmente maneja todas las funciones relacionadas con la mayoría de las reglas del negocio y los recursos de datos.

Las funciones que lleva a cabo el proceso servidor se resumen en los siguientes puntos:

• Aceptar los requerimientos de bases de datos que hacen los clientes.

• Procesar requerimientos de bases de datos.

• Formatear datos para trasmitirlos a los clientes.

[2] comprende los componentes que procesan la salida del front-end

• Procesar la lógica de la aplicación y realizar validaciones a nivel de bases de datos.

2.4.3 SOCKETS en C SHARP

Socket es un concepto abstracto por el cual dos programas (posiblemente situados en computadoras distintas) pueden intercambiarse cualquier flujo de datos, generalmente de manera fiable y ordenada por lo que un socket puede estar definido por una dirección IP, un protocolo y un número de puerto.

Es decir, es un método para que aplicaciones situadas en distintos ordenadores (o no, pueden comunicarse perfectamente aplicaciones situadas en el mismo ordenador) puedan comunicarse. Para ello, se necesita la IP, un Puerto y un Protocolo.

En C Sharp la clase **Socket** proporciona un extenso conjunto de métodos y propiedades para las comunicaciones en red. La clase Socket permite realizar transferencias de datos sincrónicas y asincrónicas mediante cualquiera de los protocolos de comunicación..

La clase Socket sigue el modelo de denominación de .NET Framework para los métodos sincrónicos. Por ejemplo, el método asincrónico *Receive* corresponde a los métodos asincrónicos *BeginReceive* y *EndReceive*.

Si la aplicación sólo requiere un subproceso durante su ejecución, es recomendable utilizar los métodos siguientes, diseñados para el modo de funcionamiento sincrónico.

Si utiliza un protocolo basado en conexiones como TCP, el servidor podrá escuchar las conexiones mediante el método *Listen*. El método *Accept* procesa las solicitudes de conexión entrantes y devuelve un Socket que puede usar para intercambiar datos con el host remoto. Se puede usar este valor devuelto de Socket para llamar al método *Send* o *Receive*. Es recondable llamar al método *Bind* antes de llamar al método Listen si se desea especificar la dirección IP local y el número de puerto. si lo que se desea es que el

proveedor de servicios subyacente le asigne un puerto libre es recomendable utilizar cero como número de puerto. Para conectar con un host de escucha, llame al método *Connect*. Para comunicar datos, se debe llamar al método *Send* o *Receive*.

3. PROCEDIMIENTOS EMPLEADOS Y ACTIVIDADES DESARROLLADAS.

Durante el desarrollo del proyecto de administración del equipo de cómputo del laboratorio central de informática se realizaron diversos procedimientos y actividades, que nos ayudaron a llegar al desarrollo del mismo. A continuación se describirán las actividades y procedimientos realizados para el desarrollo de este proyecto.

3.1 Análisis de alternativas

Primeramente se realizó un análisis para determinar las herramientas que se podrían utilizar para el desarrollo del proyecto. Este análisis consistió en una investigación en internet, libros y maestros que laboran en la UNISON.

Una vez analizadas cada una de las alternativas se decidió llevar a cabo la alternativa de utilizar la herramienta de desarrollo Visual Studio 2010 con el lenguaje de programación por motivos del tiempo de desarrollo y términos de una mejor usabilidad.

Ya que la alternativa de desarrollar el proyecto en lenguaje visual basic o visual c++ con la herramienta de visual estudio se descartaron por los pocos conocimientos que tenía de estos lenguajes.

En seguimiento se realizó una investigación sobre la estructura, desarrollo, ventajas y desventajas de aplicaciones cliente-servidor. En la investigación se obtuvo información de algunos conceptos esenciales para las aplicaciones cliente-servidor en CSHARP.

3.2 Conceptos fundamentales y Actividades

A continuación se muestran algunos conceptos fundamentales y actividades desarrolladas para el funcionamiento de las aplicaciones cliente servidor en C#:

- Aplicaciones cliente-servidor.
- Redes LAN.
- Modelo de aplicación distribuida para cliente servidor.
- Librerías, clases y métodos cliente servidor en CSHARP.
- Serialización.
- Windows Forms.
- Uso de DLLimports y las APIs de Windows.
- Timers CSHARP.
- Hacer ping a las máquinas clientes.
- Asignar IP manualmente a máquinas cliente.

3.3 Actividades desarrolladas y pruebas

Una vez estudiadas cada uno de los conceptos, actividades y características requeridas para el proyecto procedí a desarrollar elementos necesarios para la realización de las aplicaciones Cliente y Servidor.
Una vez desarrolladas las aplicaciones mencionadas en este proyecto, la sucesión a esto fue utilizar un modem y cables de red para establecer una pequeña red local LAN para conectar en este caso mi computadora portátil (Servidor) y dos computadoras de escritorio para establecer la comunicación en red.
Una vez conectadas las computadoras tanto de escritorio (Clientes) como mi computadora portátil procedí a ejecutar el comando ping del MS DOS para comprobar la visibilidad en red de todos los equipos.
Debido a que el primer intento falló la ejecución del comando ping ya que no se encontraban visibles las direcciones IP de las maquinas Cliente procedí a

asignarles direcciones IP manualmente dando como resultado una visibilidad correcta entre todos los equipos de cómputo.

3.3.1 Prueba 1 de funcionalidad y conexión

Después inicié el proceso de compilación de la Aplicación Cliente estableciendo primeramente como dirección IP a conectar la de mi computadora portátil (servidor). Posteriormente procedí a buscar el archivo ejecutable Cliente.exe en la carpeta llamada *bin* del directorio de proyecto Visual CSharp e hice una copia de la aplicación Cliente y la ejecuté en cada una de las Maquinas Cliente dando como resulta la siguiente imagen:

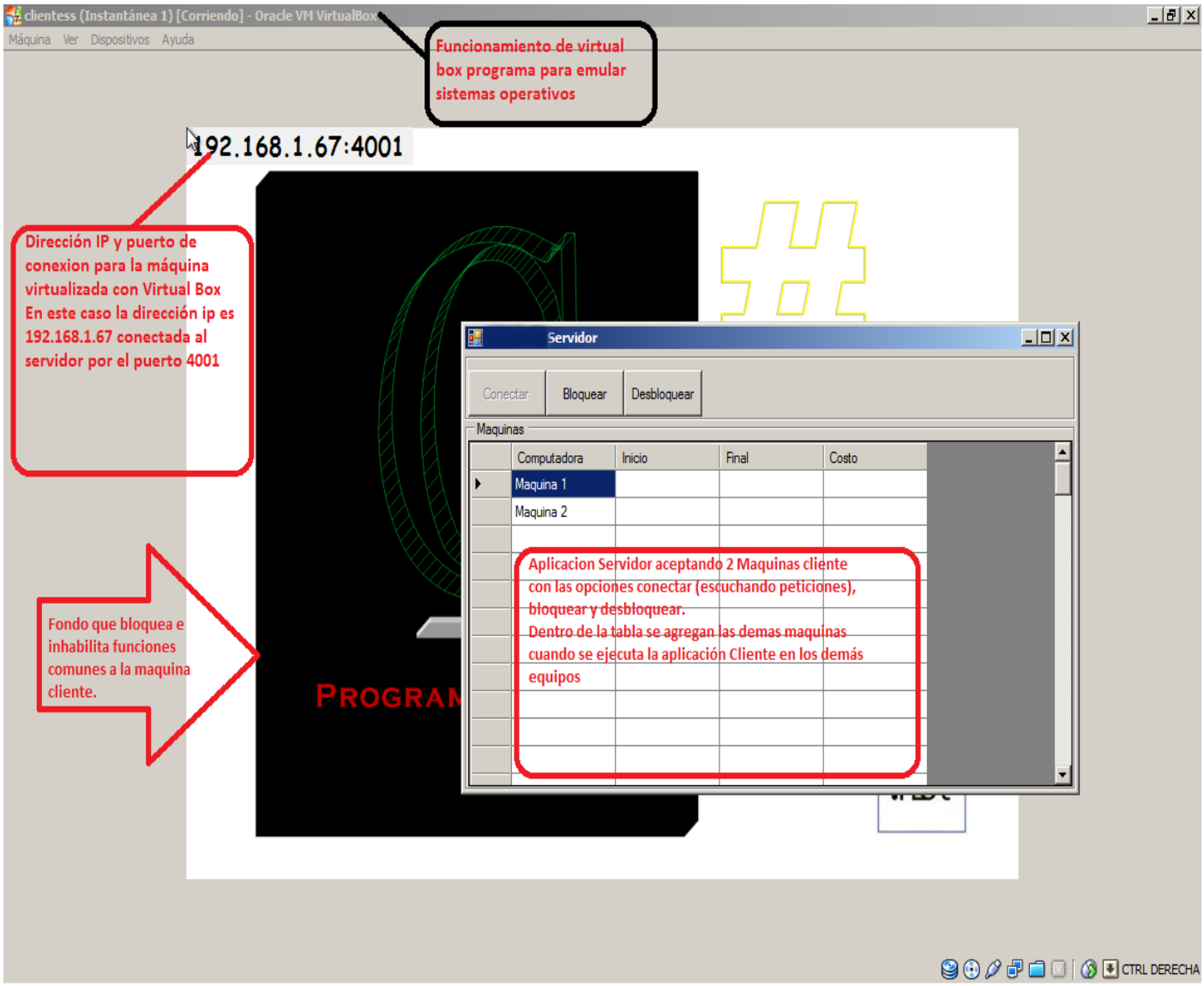


Figura 3.1 Clientes conectados y bloqueados por Servidor

En la Figura 3.1 se puede apreciar una maquina Cliente bloqueada por la aplicación Cliente ejecutándose y al mismo tiempo virtualizada por el programa Virtual Box el cual ayuda virtualizar sistemas operativos y en consecuencia se pudo tener una maquina Cliente si las limitaciones físicas de tener que obtener equipos de cómputo para pruebas.

En la imagen se puede ver en la parte superior corriendo un sistema operativo gracias a Oracle VM Virtual Box por lo que había una maquina Cliente lista para la implementación de la aplicación Cliente. También se puede apreciar superior izquierda una serie de números los cuales son la dirección IP de mi computadora portátil o Servidor. Y también se puede ver el número de puerto por el cual se ha estableció la conexión hacia el servidor.

La ventana llamada Servidor muestra a una maquina Cliente conectada (Maquina 1) y otra computadora de escritorio del CSTI. Los dos equipos conectados en la imagen están bloqueados por el servidor porque lo están inhabilitadas las funciones comunes o básicas temporalmente para trabajar del sistema operativo, donde se ejecutaron las aplicaciones Cliente.

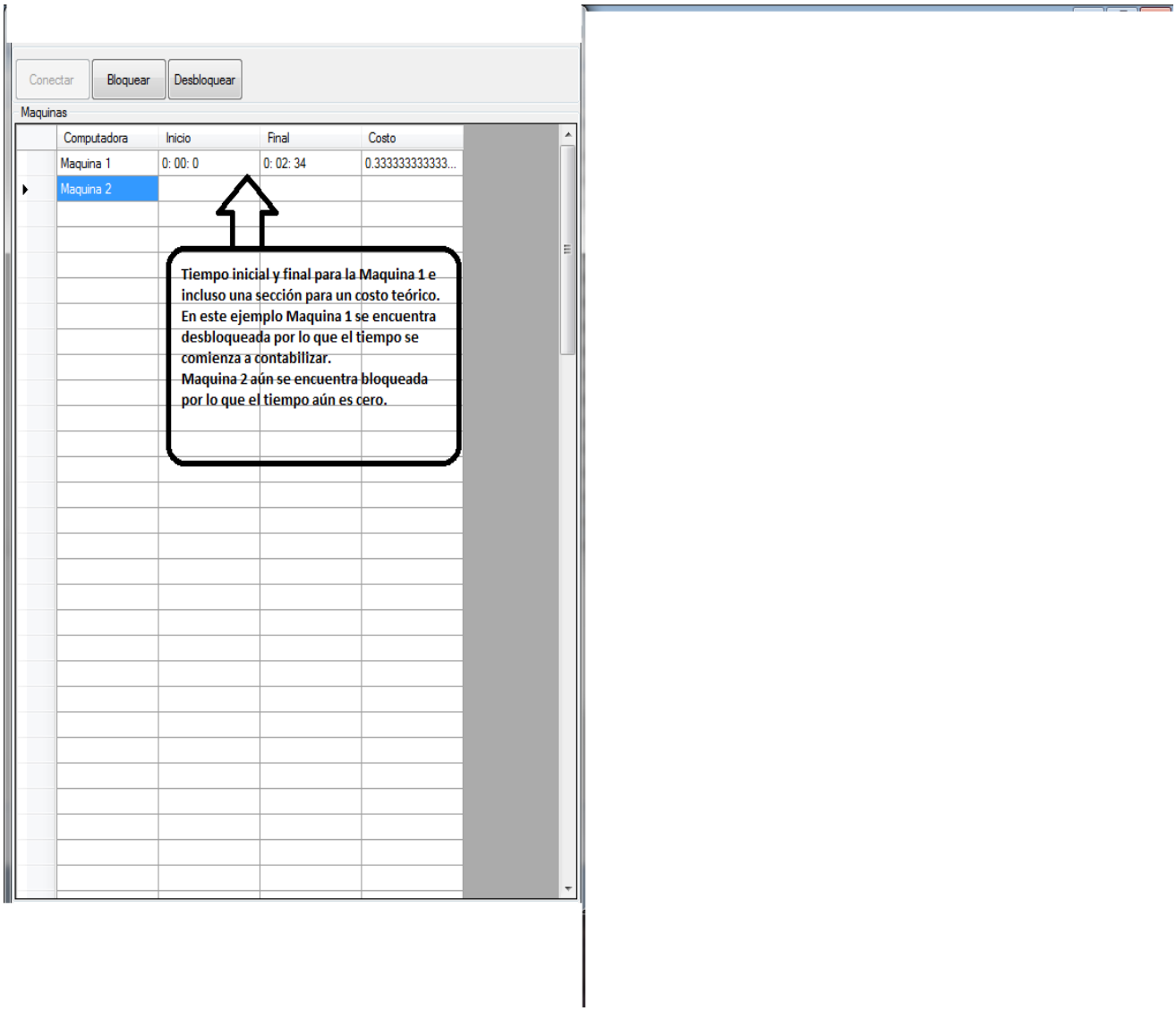


Figura 3.2. Cliente o Maquina 1 desbloqueado

3.3.2 Prueba 2 de funcionalidad y conexión.

En la figura 3.2 se puede ver a Maquina 1 con el tiempo final de 2:34 debido a que se seleccionó la dicha máquina y se oprimió el botón Desbloquear y en consecuencia de ello inició la contabilización del tiempo. Nótese que en Maquina 2 el tiempo aún permanece en cero ya que la aún no se ha desbloqueado por lo que permanecerá de esta manera hasta que repita el mismo proceso que tuvo Maquina 1.

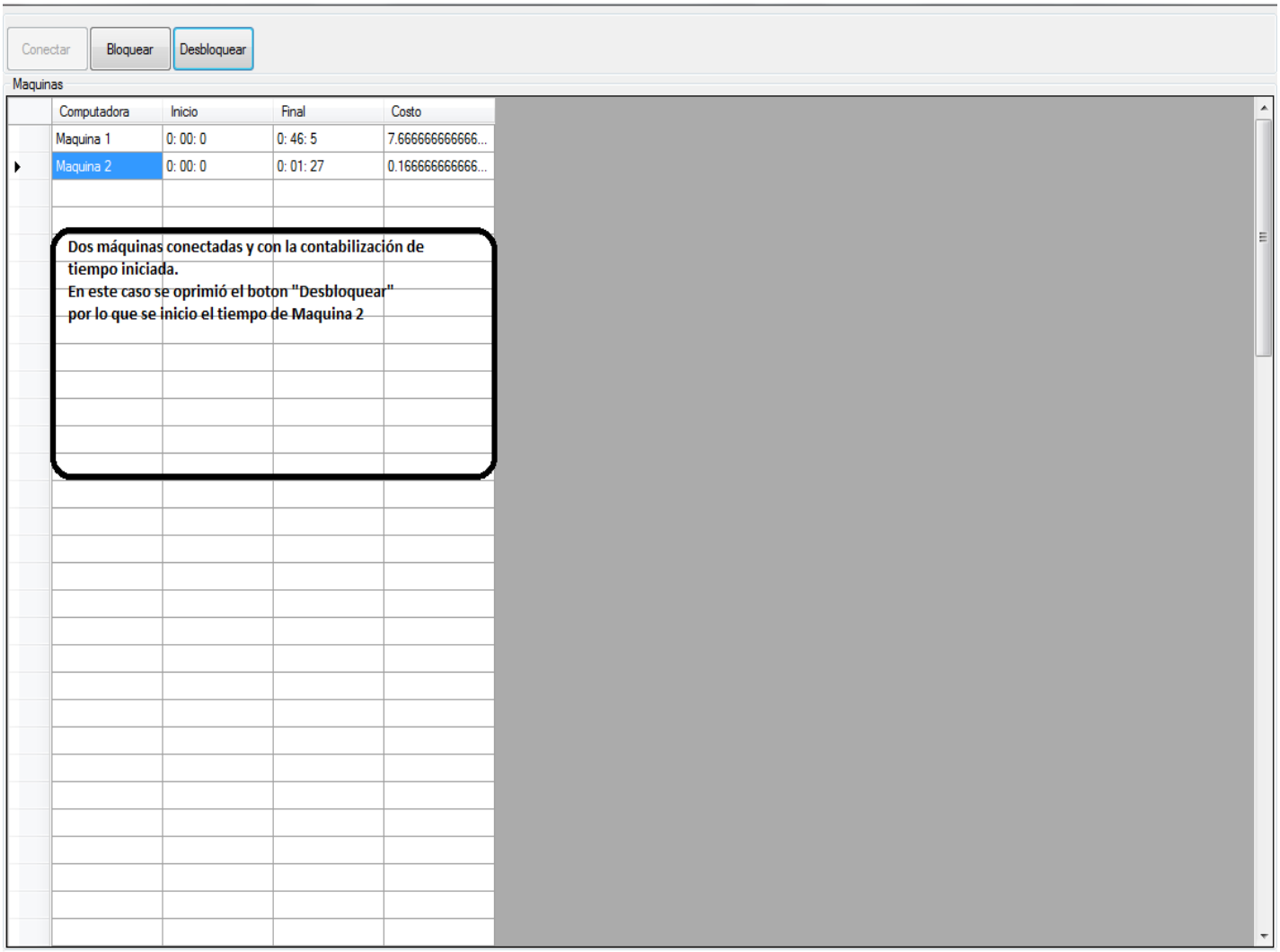


Figura 3.3 Dos Clientes desbloqueados

3.3.3 Prueba 3 de funcionalidad y conexión

En la figura 3.3 se puede ver como ahora las dos Máquinas para la primera prueba están conectadas, desbloqueadas y con la contabilización del tiempo iniciada. Partiendo de la base de que la primera prueba para el cliente servidor tuvo éxito procedí a realizar pruebas bajo otras condiciones.
Dejé en conexión durante más de una hora 4 Maquinas pero solo desbloqueando la

primera para iniciar el conteo del tiempo.

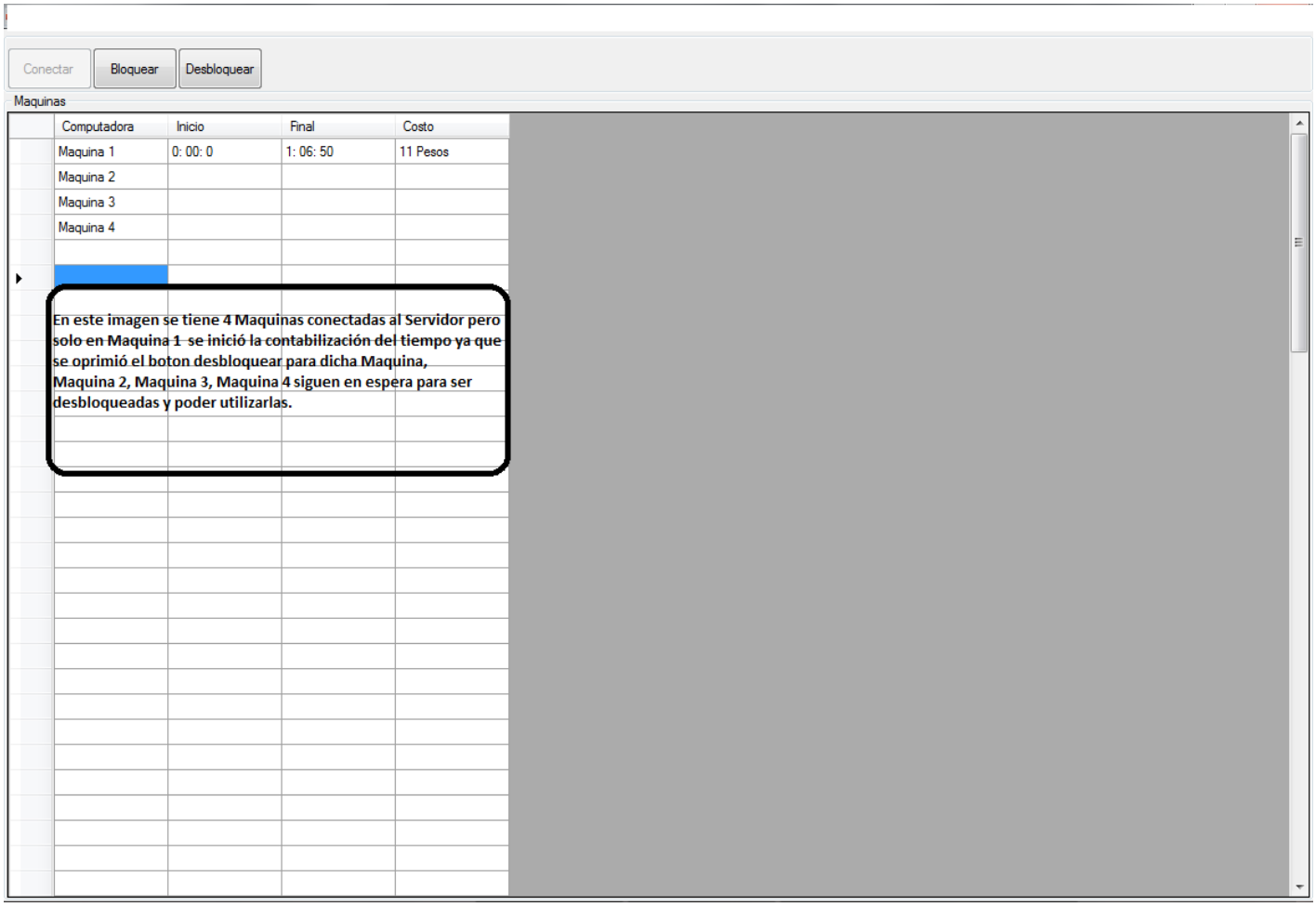


Figura 3.4 Cuatro Clientes Conectados

En la figura 3.4 como se mencionó anteriormente establecí conexión durante más de una hora a 4 Maquinas Cliente pero sola la primer Maquina desbloqueada. Dando como resultado una prueba exitosa y sin pérdida de conexión dado que cada máquina Cliente utiliza diferentes puertos para la conexión con el Servidor.

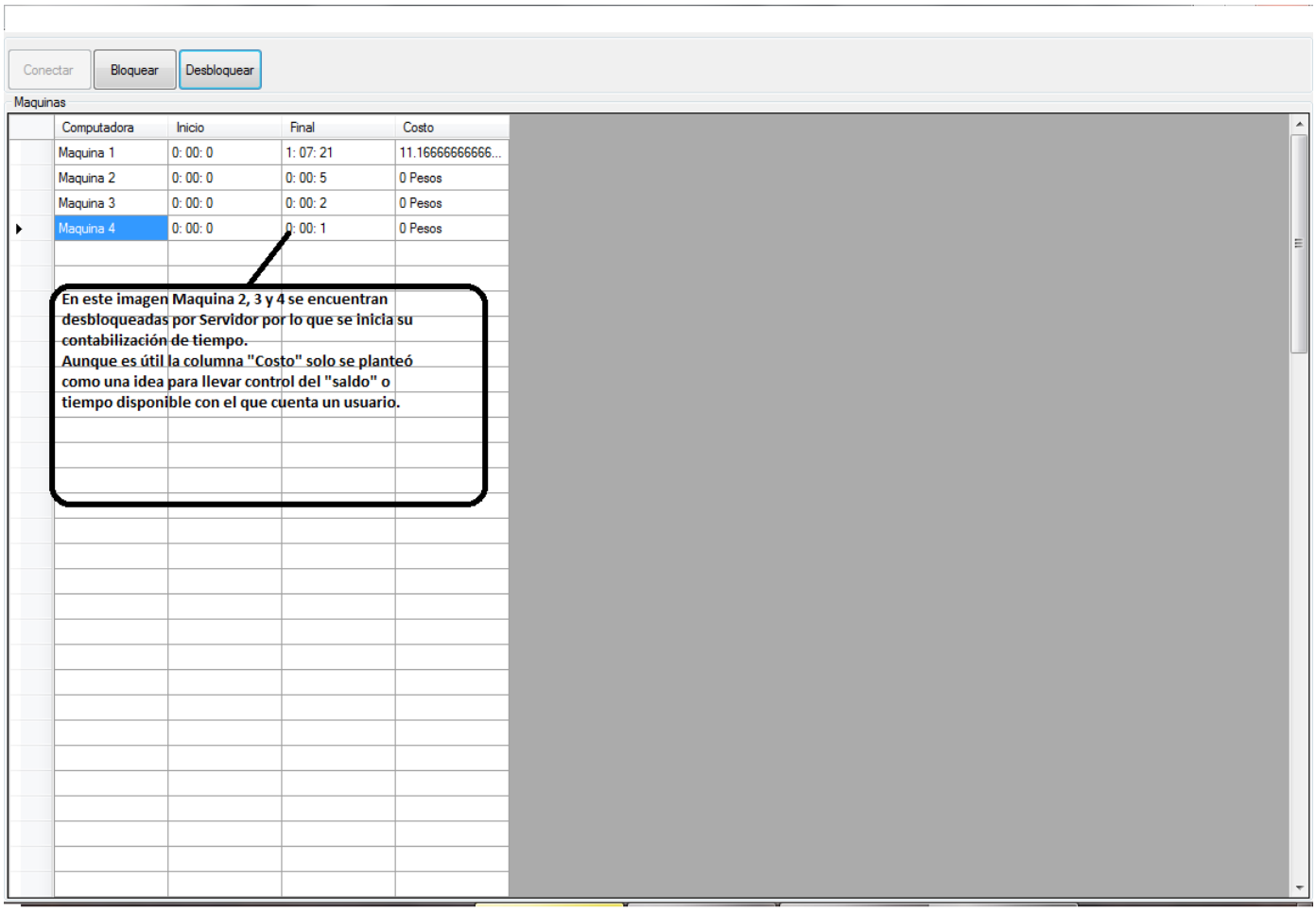


Figura 3.5 Cuatro Clientes conectados y desbloqueados-

3.3.4 Última prueba de funcionalidad y conexión

Por último procedí a desbloquear el resto de las Maquinas Cliente para iniciar el proceso de contabilización de tiempo para cada una. Adicionalmente agregué una opción para la contabilización del tiempo o saldo de los clientes, en este caso para los alumnos se haría una mejor referencia como el saldo o las horas disponibles para uso de equipos de cómputo.

En anexo 1 viene el código documentado de la aplicación Cliente

4. RESULTADOS OBTENIDOS

El resultado final de mis prácticas profesionales fue la entrega de las aplicaciones funcionales y documentadas Cliente Servidor en CD al Proyecto de Administración del Equipo de Cómputo del LCI para su posterior uso con la base de datos de los alumnos de la Universidad de Sonora. Además de que dio una capacitación a los miembros involucrados en el proyecto para manejo de las aplicaciones y el código documentado posibles modificaciones.

Además como parte adicional a mis practicas impartí un pequeño curso de Google Maps API v3 que me fue solicitado por mi tutor de prácticas M.C. Jorge Franco Romero Aguilar para algunos miembros del CSTI.

5. Conclusiones y recomendaciones.

Como conclusión puedo decir que en estos momentos de formación profesional fue un gran acierto haber realizado mis prácticas profesionales en el Centro de Servicios y Tecnologías de Información debido a que pude aplicar gran parte de los conocimientos y habilidades aprendidos durante la carrera.

Además de que pude aprender a trabajar en equipo ya que el proyecto en el que realicé mis prácticas comprendía varias partes entre ellas la que me fue asignada.

También adquirí conocimientos y desarrollé habilidades nuevas de análisis de sistemas debido a que pude aplicar conocimientos de análisis, diseño y desarrollo de sistemas de información.

Mi recomendación es que los alumnos de Ingeniería en sistemas información deberían entrar a unos de los tantos proyectos que gestiona el Centro de Servicios y Tecnologías de Información ya que la experiencia adquirida es de gran ayuda para afrontar nuevos retos. Y puesto que la formación académica es de vital importancia para los alumnos, mi recomendación es que el alumno debe valorar los conocimientos y oportunidades que se le presentan durante su estadía en la carrera ya que sin ellos sería difícil absorber los nuevos conocimientos y habilidades que depare el futuro.

6. Fortalezas y debilidades que el alumno experimentó al realizar la práctica profesional, relacionadas con los conocimientos, actitudes, y habilidades adquiridos durante sus estudios en la Universidad de Sonora.

Durante mi estancia profesional dentro del proyecto de Administración del Equipo de Cómputo del LCI, se me otorgaron tareas donde mis habilidades adquiridas con mis estudios dentro de la Universidad de Sonora no fueron suficientes para satisfacer dichas tareas.

Las debilidades que se me presentaron más comúnmente fue, no contar con los conocimientos suficientes para trabajar la parte del proyecto que me fue asignada. Algunas de las debilidades presentadas fueron no haber obtenido los conocimientos de trabajar con herramientas de programación basadas en .NET o no haber profundizado en los diferentes temas en específicos clientes y servidores.

Otra debilidad desde mi punto de vista fueron los pocos conocimientos prácticos del área de redes con los que contaba. Y debido a esto puedo decir que obtuve muchas fortalezas y habilidades que probablemente de otro forma hubiera adquirido hasta que me hubiese desempeñado en el ambiente laboral.

De forma más explícita puedo mencionar que apliqué conocimientos de análisis y diseño de sistemas de información. Estructura de datos en lo referente a la programación y conocimientos de redes computacionales, además de entender el funcionamiento y estructura de las aplicaciones cliente-servidor.

7. OPORTUNIDADES DETECTADAS DURANTE MIS PRÁCTICAS

Para finalizar puedo mencionar a que gracias a que realicé mis practicas dentro de la Universidad de Sonora pude desenvolverme en mi área y trabajar equipo afectando positivamente directamente a mi comportamiento de mi futuro ambiente laboral desempeñándome mejor en lo que hago siendo autodidacta y no dependiente de alguien más para adquirir conocimientos nuevos.

8. REFERENCIAS BIBLIOGRÁFICAS

Páginas de internet

1.-Sitio web de programación

http://cspromex.activo.mx/

2.-Sitio web oficial de Microsoft para la documentación de C sharp

http://msdn.microsoft.com/es-es/library/ms269115(v=vs.80).aspx

3.-Sitio web de programación en inglés

http://stackoverflow.com/

Libros

4.- Diego Ruiz, C#: La Guia Total Del Programador, Mp, 2005, 390 páginas

5.-Harvey M. Deitel,Como Programar En c# De Deitel, 2007, 1080 páginas

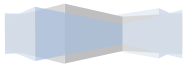
9. ANEXOS

Anexo 1

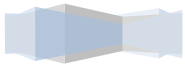
CODIGO DOCUMENTADO DE APLICACIÓN CLIENTE

```
using System;
using System.Runtime.InteropServices;//para llamar a los DLL y así usar los
metodos externos o de Windows
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Net;
using System.Net.Sockets;//envio y recepción de datos

namespace CiberCafe_Cliente
{
    public partial class Form1 : Form
    {
        public static TcpClient Cliente = new TcpClient();//se instancia
TcpClient para usar conexiones TCP
        public static String Recibido = "";//variable String cuyo valor inicial
es ""
        public Form1()
        {
```

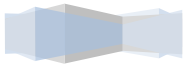
```
            MessageBox.Show("Form1");
            InitializeComponent();
        }
        #region Imports
        //Constantes para ShowWindow
        private const int ESCONDER = 0;
        private const int MOSTRAR = 1;

        //Busca el handle de una ventana hija a partir de su Hwnd Parent y el
nombre de clase
        [DllImport("user32")]
        private static extern IntPtr FindWindowEx(IntPtr hWnd1, int hWnd2, string
lpsz1, string lpsz2);

        //Busca el HWND de la ventana, en este caso la de la barra de tareas de
windows ( mediante el nombre de clase Shell_TrayWnd )
        [DllImport("user32")]
        private static extern IntPtr FindWindow(string lpClassName, string
lpWindowName);

        //Oculta y muestra una ventana a partir de su HWND
        [DllImport("user32")]
        private static extern void ShowWindow(IntPtr hwnd, int nCmdShow);
        //regresa o matiene al puntero del Mouse en posicion x, y
        [DllImport("User32.dll")]
        public static extern bool SetCursorPos(int x, int y);
        //contructores para para bloquear administrador de tareas[1], barra de
notificaciones[2], iconos[3], reloj[4] e inicio de Windows[5]
        private IntPtr HWND_TaskBar;//1
        private IntPtr HWND_TrayNotify;//2
        private IntPtr HWND_Iconos;//3
        private IntPtr HWND_Reloj;//4
        private IntPtr HWND_Inicio;//5

        #endregion

        private void tmConexionServidor_Tick(object sender, EventArgs e)//timer
que sirve para establecer la conexion al Servidor
        {

            try
            {
                if (Cliente != null && Cliente.Connected == false)//chea si el
Cliente esta conectado
                {
                    Cliente.Connect(IPAddress.Parse("192.168.1.143"),
4000);//intenta conexión con la ip especificada con su correspondiente puerto
                    lblEstado.Text = "Conectando con el servicio...";//actualiza
el lblEstado.Text con el String dado
                }
                else
                {
                    if (Cliente.Available > 0)//valida si mas de un cliente
conectado o hay datos
                    {
                        Recibido = ER.Recibir(Cliente);/*la variable static
Recibido de tipo String recibe
                        recibe atravez de las clase ER con el metodo Recibir() al
Cliente*/

                        lblEstado.Text = Recibido;//el etiqueta del label
lblEstado.Text se actualiza a el contenido de la variable static Recibido
```

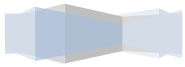
```
                        Cliente.Close();//cierra la instancia Cliente para
intentar una nueva
                        Cliente = new TcpClient();//inicializa una nueva
instancia Cliente de tipo TcpCliente reconectar con el Servidor
                        tmConexionServidor.Enabled = false; //temporizador
tmConexionServidor se establece como falso ya que aun esta intentando conectar
                        tmConexionCliente.Enabled = true;//el temporizador
tmConexionCliente se establece como true para seguir intentado la conexion
                    }
                }
            }
            catch (Exception)//en caso que no se conectara se encapsula la
excepcion
            {
                lblEstado.Text = "Error de conexion..Reintentando.."; //<- y se
actualiza el label a el string dado
                Cliente = new TcpClient(); //inicia un nuevo Cliente para
reintentar conexion
            }

        }

        private void tmConexionCliente_Tick(object sender, EventArgs e)//timer
para conexion de cliente
        {
           // MessageBox.Show("tmConexionCliente_Tick");
            try
            {
                if (Cliente != null && Cliente.Connected == false)/*verifica que
dos posible condicones se cumplan
                                                                  * 1.-Cliente
sea diferente a Null
                                                                  * 2.-Cliente
esta conectado */
                {
                    Cliente.Connect(IPAddress.Parse("192.168.1.143"),//conecta al
Cliente con la dirección IP (192.168.1.2)
                        Convert.ToInt32(Recibido));//convierte la variable static
String Recibido a Entero
                    lblEstado.Text =
Cliente.Client.RemoteEndPoint.ToString();//actualiza el label lbEstdo.Text aL
socket Client

                }
                else
                {
                    if (Cliente.Available > 0)//si al menos un cliente esta
conectado
                    {
                        Recibido = ER.Recibir(Cliente);//la clase ER utiliza el
metodo String Recibir para el string BB o DD para bloqueo y desbloqueo de Cliente

                        if (Recibido == "BB") //en caso de BB
                        {
                         Bloquear();//se llama al metodo bloquear
                        }
                        else if (Recibido.StartsWith("DD"))//en caso de DD
                        {
                            Desbloquear();//es llamado el metodo Desbloquar
                        }
                        else
                        {
```

```
                            Cliente.Close(); //elimina la instancia Cliente sin
cerrar la conexion para
                            Cliente = new TcpClient();//instanciar de nuevo y con
que el mismo nombre Cliente
                            tmConexionServidor.Enabled = true;//estable el timer
tmConexionServidor como verdadero
                            tmCursor.Enabled = false;
                            tmTiempo.Enabled = false;
                            tmConexionCliente.Enabled = false;
                        }
                    }
                }
            }
            catch (Exception)//se obtiene la excepcion general en caso de una
falla en la conexion
            {
                lblEstado.Text = "Error de conexion 1";//actualiza LblEstado.Text
con el string dado
                tmConexionServidor.Enabled = true;
                tmConexionCliente.Enabled = false;
                Cliente = new TcpClient();//en caso de fallo de conexion se hace
una instancia de TcpClient
            }
        }

        private void Form1_FormClosing(object sender, FormClosingEventArgs e)
        {
            MessageBox.Show("Form1_FormClosing");
            if (Cliente.Connected == true)
            {
                ER.Enviar(Cliente, "SS");/*cuando se cierra el Form Cliente envia
el string SS
                                          * para removerlo  del datagridvied en
                                           aplicacion cliente*/
                Cliente.Close();//elimina la instancia de Cliente
            }
        }


        private void Form1_Load(object sender, EventArgs e)
        {
            MessageBox.Show("Form1_Load");
            tmConexionServidor.Enabled = true;
            //busca todas las apis de windows para cargarlas
            HWND_TaskBar = FindWindow("Shell_TrayWnd", null);
            HWND_Inicio = FindWindow("BUTTON", null);
            HWND_TrayNotify = FindWindowEx(HWND_TaskBar, 0, "TrayNotifyWnd",
null);
            HWND_Iconos = FindWindowEx(HWND_TrayNotify, 0, "Syspager", null);
            HWND_Reloj = FindWindowEx(HWND_TrayNotify, 0, "TrayClockWClass",
null);

            this.SetBounds(0, 0, Screen.PrimaryScreen.Bounds.Width,
Screen.PrimaryScreen.Bounds.Height);
            this.BackgroundImage = Properties.Resources.Logo3;
            this.BackgroundImageLayout = ImageLayout.Stretch;
            Bloquear();
        }
        //variables globales
        int segundos, minutos, horas;//variables de tipo int
        string hora;//variable string hora que contendra los valores de segundos
minutos y segundos
```

```
        bool ban = true;
        string MIN;
        private void tmrTiempo_Tick(object sender, EventArgs e)//timer tmrTiempo
para contar tiempo de cliente
        {
            if (minutos < 10)//
            {
                MIN = "0" + minutos.ToString();//MIN sera igual a la suma de 0 +
minutos=(inicialmente 0)
            }
            else {
                MIN = minutos.ToString();
            }
            hora = horas.ToString() + ": " + MIN+ ": " +
segundos.ToString();//hora contiene las horas, minutos y segundos contados
            lblEstado.Text = hora;//muesta el tiempo en el Label
            segundos++;//aumenta en 1 el valor de la variable segundos
            if (segundos > 59) {
                minutos++;//aumenta en 1 a minutos cada vez que sea mayor a 59
segudos
                segundos = 0;//una vez que aumenta en 1 a minutos ...segundos se
vuelve 0 para volver a contar
            }
            if (minutos > 59)//si han pasado 59 minutos la variable int horas
aumenta en 1 su valor
            {
                minutos = 0;//minutos vuelva a 0 para volver a contar los minutos
una vez que concluyo la hora
                horas++;//aumenta en 1 el valor de la variables horas
            }
            if (horas > 23)//si la variable horas es 23 significa que han pasado
24 horas y la variable hora será igual a 0 para volver a contar
            {
                horas = 0;//en caso de que haya pasado un dia
            }
            if (ban)//condicion verdadera mientras corre la aplicacion ya que la
variable bool ban esta declarada true
            {

                ER.Enviar(Cliente, "TI" + hora);/*si la condicion es verdadera se
hace uso de la clase la cual utiliza el metodo Enviar
                                                el cual requiere dos parametros
que son la variable de tipo TcpClient y la variable
                                                * hora la cual contiene el
timepo en horas minutos y segundos del cliente todo
                                                esto para el tiempo inicial TI
claro*/

                ban = false;
            }
            else {
            ER.Enviar(Cliente, "TF" + hora);/*en caso de que se cumpla la
codicion se actua de la misma forma con la misma clase
                                            * y metodos solo que se envia los
datos del mismo Cliente y ahora se envia el tiempo final TF*/
            }
        }
        private void tmrCursor_Tick(object sender, EventArgs e)
        { SetCursorPos(0, 0);
            //estable el cursos en la posiciona dada para que no se pueda mover
de ahí
        }
```

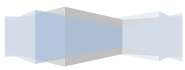



```
        #region Metodos

        public void Bloquear()
        {
            this.SetBounds(0, 0, Screen.PrimaryScreen.Bounds.Width,
Screen.PrimaryScreen.Bounds.Height);//establece una imagen al tamaño de pantalla
            this.TopMost = true;//estable el formulario de nivel superior
            ShowWindow(HWND_TaskBar, ESCONDER);//esconde el administrador de
tareas
            ShowWindow(HWND_Iconos, ESCONDER);//esconde los iconos
            ShowWindow(HWND_Inicio, ESCONDER);//esconde el boton de inicio
            ShowWindow(HWND_TrayNotify, ESCONDER);//esconde la barra de
notificaciones
            ShowWindow(HWND_Reloj, MOSTRAR);//esconde el reloj
            tmTiempo.Enabled = false;//deshabilita o establece el timer tmTiempo
            tmCursor.Enabled = true;//habilita o establece al puntero para
mantener la posicion 0,0
            lblEstado.Text = "";
            //mientras este bloqueada la maquina cliente el tiempo no debe correr
por lo que se deben igualar a cero las siguientes variables:
            minutos = 0;
            segundos = 0;
            horas = 0;
            ban = true;
            this.BackgroundImageLayout = ImageLayout.Stretch;//obtiene o
establece el diseño de la imagen de fondo definida por el tamaño total del
rectangulo
            this.BackgroundImage = Properties.Resources.Logo3;//obtiene la imagen
de fonde para mostrarla
        }

        public void Desbloquear()
        {
            this.TopMost = false;//establece como falso si debe mostrarse el
formulario como nivel superior
            this.SetBounds(0, 0, Screen.PrimaryScreen.Bounds.Width, 30);//estable
el formulario con ancho al tamaño de pantalla y alto de 30
            ShowWindow(HWND_TaskBar, MOSTRAR);//muestra el administrador de
tareas
            ShowWindow(HWND_Inicio, MOSTRAR);//muestra el boton de inicio
            ShowWindow(HWND_Iconos, MOSTRAR);//muestra los iconos
            ShowWindow(HWND_TrayNotify, MOSTRAR);//muestra la barra de
notificaciones
            ShowWindow(HWND_Reloj, MOSTRAR);//muestra el reloj
            tmTiempo.Enabled = true;//empieza a contar el timepo estableciendo
tmTiempo como true
            tmCursor.Enabled = false;//permite que se pueda mover el cursor
estableciendolo como false
            this.BackgroundImage = null; //inhabilita el fondo de pantalla
        }
        #endregion


    }
}
```


ANEXO 2

CODIGO DOCUMENTADO DE LA APLICACIÓN SERVIDOR

```
using System;
using System.Text;
using System.Net;
using System.Net.Sockets;
using System.IO;
using System.Windows.Forms;

namespace CiberCafe_Servidor
{
    class ER//nombre de la clase que utiliza el la aplicacion Servidor
    {
        public static void Enviar(TcpClient cliente, String Dato) {
            try
            {
                NetworkStream netstream = cliente.GetStream();
                StreamWriter escribir = new StreamWriter(netstream);
                escribir.WriteLine(Dato);
                escribir.Flush();
            }
            catch (Exception) { MessageBox.Show("Error al enviar no hay socket");
}
        }
        public static void Enviar(TcpClient cliente, Object Dato)
        {
            NetworkStream netstream = cliente.GetStream();
            StreamWriter escribir = new StreamWriter(netstream);
            escribir.WriteLine(Dato);
            escribir.Flush();
        }

        public static String Recibir(TcpClient cliente) {
            try
            {
                NetworkStream netstream = cliente.GetStream();
                StreamReader leer = new StreamReader(netstream);

                return leer.ReadLine();
            }
            catch (Exception) { MessageBox.Show("Error al recibir"); return ""; }
        }

        public static void Recibir(TcpClient cliente,ref object objeto)
        {
            NetworkStream netstream = cliente.GetStream();
            StreamReader leer = new StreamReader(netstream);
            objeto= leer.ReadLine();
        }

        public static void Recibir(TcpClient cliente,ref String Dato)
        {
            NetworkStream netstream = cliente.GetStream();
            StreamReader leer = new StreamReader(netstream);
            Dato= leer.ReadLine();
        }



    }
}
```

Carta de finalización

EL SABER DE MIS HIJOS
HARÁ MI GRANDEZA

UNIVERSIDAD DE SONORA
DIVISION DE INGENIERIA
Departamento de Ingeniería Industrial
Ingeniería en Sistemas de Información

Centro de Servicios de TI

Hermosillo, Sonora a 05 de marzo de 2013

Dr. Mario Barcelo Valenzuela
Coordinador de Practicas Profesionales
del Ingeniería en Sistemas de Información.

Por medio de la presente reciba un saludo a la vez que le informo que el alumno: MANUEL CELESTINO AGUILAR OSUNA con número de expediente: 208283119, concluyo satisfactoriamente las practicas profesionales, presentando un total de 395 horas de trabajo, en actividades correspondientes a la gestión de proyectos del Centro de Servicios de TI, correspondiente en su mayoría al proyecto de: " Sistema de Administración de Equipo de Computo para el Laboratorio de Computo Central de la Universidad de Sonora".

Agradezco de antemano su atención.

Atentamente

M.C. Jorge Franco Romero Aguilar
Coordinador del Centro de Servicios de TI

EL SABER DE MIS HIJOS
HARÁ MI GRANDEZA
DEPARTAMENTO DE
INGENIERÍA INDUSTRIAL

Formato de terminación de practicas

Universidad de Sonora
Departamento de Ingeniería Industrial
Coordinación de Prácticas Profesionales

FPP-3
Formato de Liberación de la Práctica Profesional

Hermosillo, Sonora, 05 (DÍA) Marzo (MES) 2013 (AÑO)

Con mi carácter de Tutor de Prácticas Profesionales, hago constar que:
I.- El alumno(a) MANUEL CELESTINO AGUILAR OSUNA,

de la carrera ING. EN SIST. DE INFORMACION **con número de expediente:** 208203119,

ha cumplido con la entrega oportuna de:
a.- Los reportes de avances periódicos de su práctica profesional.
b.- El informe técnico del proyecto realizado.

II.- He corroborado que los contenidos y tiempos de los reportes de avances están acordes con lo planeado en los anexos del formato de inscripción FPP-1, y que los contenidos y forma del informe técnico satisfacen los requerimientos especificados en la normatividad actual.

III.- El número de horas acumuladas de práctica profesional, de acuerdo a los reportes de avances, es de 385

Por lo anteriormente expuesto, no tengo inconveniente alguno en dar por liberado(a), al (la) alumno(a), anteriormente referido(a), del cumplimiento de la práctica profesional:

[×] **TOTALMENTE, y evaluarlo(a) con 20 créditos cumplidos.**

IV.- Debido a que el alumno no pudo terminar su práctica profesional en la empresa asignada, con base en sus reportes, y dado que ha acumulado [ ] **horas de práctica, no tengo inconveniente alguno en dar por liberado(a), al (la) alumno(a), anteriormente referido(a), del cumplimiento de la práctica profesional:**

[ ] **PARCIALMENTE, y evaluarlo(a) con los siguientes créditos:**

Con número [ ] **Con letra** [ ]

MOTIVOS POR NO HABER TERMINADO CON LA PRÁCTICA PROFESIONAL

ATENTAMENTE:
Nombre y firma del Tutor de Prácticas Profesionales

MC. JORGE F. ROMERO AGUILAR

Vo. Bo.
Nombre y Firma del Coordinador de Prácticas Profesionales

EL SABER DE MIS HIJOS HARÁ MI GRANDEZA
DEPARTAMENTO DE INGENIERÍA INDUSTRIAL

Original: Coordinación de Prácticas Profesionales
C.c.p.: Alumno
C.c.p.: Tutor de Prácticas Profesionales