80P

Universidad Técnica Particular de Loja
Biblioteca General
Recibido el 2003-08-26
Valor $ 1=
Nº Clasificación 2003 P695 I. In. 20

005

Hardware
software
Red windows
UPSI - UTPL

005.6
005

005 X 862 DL-

005.456

UNIVERSIDAD TÉCNICA PARTICULAR DE LOJA

ESCUELA DE SISTEMAS INFORMATICOS Y CIENCIAS DE LA COMPUTACIÓN

TITULO DEL PROYECTO

"HERRAMIENTA PARA MONITOREAR LOS RECURSOS HARDWARE Y SOFTWARE EN UNA RED WINDOWS, EN LA UPSI DE LA UTPL."

Trabajo presentado como requisito para optar por el Título de Ingeniero en la especialización de Informática

AUTOR:

CARLOS E. PIZARRO TORRES

ASESOR:

Ing. Juan Carlos Torres D.

LOJA - ECUADOR

AÑO – 2.003

CERTIFICACIÓN

Ing. Juan Carlos Torres D.
DIRECTOR DE TESIS

CERTIFICA:

Que el presente trabajo de investigación, ha sido elaborado íntegramente por el autor y revisado minuciosamente por el suscrito, por lo que me permito autorizar su presentación.

Loja, mayo 21 de 2003

Ing. Juan Carlos Torres D.
DIRECTOR DE TESIS

CESION DE DERECHO

Yo, Carlos Eloy Pizarro Torres declaro conocer y aceptar la disposición del Art. 67 del Estatuto Orgánico de la Universidad Técnica Particular de Loja que en su parte pertinente textualmente dice: "Forman parte del patrimonio de la Universidad la propiedad intelectual de investigaciones, trabajos científicos o técnicos y tesis de grado que se realicen a través, o con el apoyo financiero, académico o institucional (operativo) de la Universidad".

f) _____

Carlos Eloy Pizarro Torres

AGRADECIMIENTO

Deseo dejar constancia de mi agradecimiento a la Escuela de Sistemas Informáticos y Ciencias de la Computación en la persona de la Ing. Liliana Enciso Directora de la misma y por su intermedio al cuerpo docente que en su momento, fueron los guías que me brindaron los primeros pasos para llegar a culminar con éxito mi carrera profesional.

Así también vaya mi agradecimiento al Ing. Juan Carlos Torres D. director de tesis y todas aquellas personas e instituciones que con buena voluntad me brindaron su ayuda para la ejecución de este trabajo.

Un especial agradecimiento a RJH por haberme dado su apoyo incondicional, así como también todos mis compañeros de trabajo.

Gracias a Todos

El Autor

AUTORIA

Las opiniones y criterios emitidos en el presente trabajo de investigación, son de absoluta responsabilidad de su autor.

Loja, mayo 21 de 2003

El autor

Carlos E. Pizarro Torres

DEDICATORIA

A mis Padres en especial a Teresita,

y todos mis familiares que me

apoyaron.

Carlos

UNIVERSIDAD TÉCNICA PARTICULAR DE LOJA
BIBLIOTECA GENERAL

TITULO

"HERRAMIENTA PARA MONITOREAR LOS RECURSOS HARDWARE Y SOFTWARE EN UNA RED WINDOWS, EN LA UPSI DE LA UTPL."

ESQUEMA DE CONTENIDOS

CERTIFICACIÓN

CESION DE DERECHO

AGRADECIMIENTO

AUTORIA

DEDICATORIA

TITULO

ESQUEMA DE CONTENIDOS

PROLOGO

INTRODUCCIÓN

OBJETIVOS

CAPITULO I.

1. DETERMINACIÓN DE LOS RECURSOS HARDWARE

1.1 Verificación de tipo de procesador

1.2 Determinación de la cantidad de memoria en el computador

1.3 Tamaño y espacio del Disco Duro

1.4 Tarjetas del Computador

1.5 Registro de datos observados

CAPITULO II.

2. MONITOREO DE LOS RECURSOS SOFTWARE

2.1 Aplicaciones en ejecución

2.2 Recursos utilizados por las aplicaciones

2.3 Liberación de recursos

CAPITULO III.

3. VERIFICACIÓN DE SOFTWARE INSTALADO

3.1 Registro del Sistema Operativo

3.2 Detección de programas instalados

3.3 Verificación de características de los programas.

CAPITULO IV

4. ANÁLISIS DEL SISTEMA

Análisis del Sistema

CONCLUSIONES

RECOMENDACIONES

ANEXOS

BIBLIOGRAFIA

INDICE

PROLOGO

Para el desarrollo de la aplicación, en primera instancia se planifico realizarla en el lenguaje de Visual Basic, pero al no existir o no tener las suficientes facilidades en el manejo de las funciones para poder realizar el acceso a todas las características de los dispositivos hardware, se opto por cambiar de herramienta de programación, y la que se eligió fue Delphi.

Mediante el nuevo lenguaje elegido (delphi) se logro realizar una programación mas profunda, llegando incluso a tener unos módulos en lenguaje ensamblador, lo que facilito la extracción de la información del hardware, lo que hace a la aplicación más independiente del sistema operativo, ya que se utiliza muy poco las funciones DLL del mismo y nos evitamos la molestia de que en algún equipo no exista una función DLL que nuestra aplicación necesita.

La información que se extrae, se la muestra finalmente en un formato XML (también permitido realizar en Delphi) lo que facilita una navegación por todos los dispositivos que se tiene y que no causa confusión al usuario final o administrador del sistema, además la administración de los datos y de los nombres de los equipos se la realizo en PowerBuilder, lo que nos facilito el trabajo y la búsqueda de datos de algún equipo determinado, así como también el manejo de las direcciones MAC de las maquinas.

INTRODUCCIÓN

El presente proyecto tiene por objetivo dar a conocer los procedimientos y formas sobre como extraer la información de un computador sin tener la necesidad de estarlo abriendo o desarmando, y así ganar mas tiempo al momento de realizar una verificación del mismo. El texto se compone de tres capítulos:

En el primero se hace referencia a los recursos hardware que posee un computador, aquí se realiza un análisis sobre como detectar un procesador, ya que existen una gran variedad de ellos y de diferente proveedor o fabricante y cada uno con diferentes versiones, además se describe cual es la estructura para obtener la información sobre la memoria tanto física como virtual, también se hace un análisis sobre el disco duro para saber que tipo y que características posee en cuanto al tamaño y espacio del mismo, por ultimo se hace una revisión general sobre las diferentes tarjetas que tiene el equipo para determinar el tipo y características de cada una de ellas.

El segundo capitulo trata sobre los recursos software del computador, como son las aplicaciones que se encuentran en ejecución en un determinado momento, aquí se detalla de que manera se las puede detectar, también se verifica los recursos que están utilizando las aplicaciones ya que cada una utiliza una cantidad determinada ya sea de memoria o de tiempo de procesador para cumplir con su tarea, también se indica la manera de liberar los recursos que esta utilizando un proceso o tarea cuando ya no deseamos que siga ocupando o consumiendo los recursos de nuestro sistema operativo.

En el tercer capitulo se hace referencia al registro del sistema operativo en el cual encontramos una descripción sobre cada una de las claves del registro interno (regedit) conque trabaja el

mismo y que es la base para su correcto funcionamiento. Un sistema operativo es el conjunto de rutinas que controlan la ejecución de las diferentes aplicaciones y que proveen la interfaz con el usuario. Este tiene tres objetivos: conveniencia (hace de la computadora una herramienta conveniente de usar), eficiencia (en el uso de los recursos del sistema), y habilidad para evolucionar (o sea, para mejorar sus componentes sin que se degrade el rendimiento del sistema); aquí también se realiza. Una descripción sobre como y donde se busca para poder detectar todo el software instalado en el equipo, ya que siempre se tiene una gran cantidad de programas que no los usamos o sabemos de la existencia de los mismos en nuestro equipo.

Para complementar todo lo que implica el texto se ha desarrollado una aplicación que nos muestra en pantalla todas las características del hardware y software que posee un equipo lo que facilitara a un administrador de un centro de computo realizar una revisión completa de todos los PCs de su lugar de trabajo sin tener que estar desarmando o abriendo cada equipo que posee.

OBJETIVOS

GENERAL

Desarrollar una Herramienta para monitorear los recursos Hardware y Software de un computador en red con S.O Windows, dentro de la Unidad de Proyectos y Sistemas Informáticos (UPSI) de la UTPL.

ESPECIFICOS

- Determinar los recursos Hardware que posee un computador que se encuentre dentro de la RED.

- Revisar el software que se encuentra instalado en cada uno de los computadores de la RED.

- Observar el rendimiento de las aplicaciones que están ejecutándose en el computador.

1. DETERMINACIÓN DE RECURSOS HARDWARE

1.1 Verificación de tipo de procesador

Los procesadores actuales, poseen una instrucción interna que se llama CPUID, que es una cadena que devuelve cierta información sobre el tipo de CPU para poder identificarlos a cada uno.

Para realizar la verificación de l tipo de CPU que posee un computador, primeramente nosotros debemos saber los tipos de cadena que devuelve la instrucción CPUID, que cada fabricante coloca en esta, a continuación se muestra un listado de estas cadenas.

El primer nivel de la instrucción CPUID retorna:

TIPO DE PROCESADOR	CADENA RETORNADA
Intel	"GenuineIntel"
AMD	"AuthenticAMD"
Cyrix	"CyrixInstead"
NexGen	"NexGenDriven"
Centaur/IDT	"CentaurHauls"
Rise	"RiseRiseRise"
UMC	"UMC UMC UMC"

El segundo nivel de la instrucción de CPUID, nos retorna información acerca del tipo, familia, revisiones (actualizaciones)

TIPO tiene estos valores:

0 – CPU primario existente en el sistema

1 - Procesadores que se reemplazan directamente en motherboards antiguos

2 – Los sistemas que utilizan un CPU secundario, o sea, son multiprocesador

La FAMILIA es casi equivalente a la generación y denota el "rendimiento":

4 - todos los 486s, AMD 5x86, Cyrix 5x86,

5 - Intel Pentium y Pentium MMX, AMD K5 y K6, Cyrix 6x86, todos los Centaur/IDT WinChip, Rise mP6

6 - El Intel Pentium Profesional, Celeron, Pentium II y Pentium III, AMD Athlon, Cyrix 6x86MX y M-II

MODELO es un número que especifica a un modelo en una familia:

por ejemplo en familia 4:

0 - i486DX

3 - i486DX2

8 - i486DX4

por ejemplo en familia 5:

2 - Pentium

4 - Pentium MMX

por ejemplo en familia 6:

1 - Pentium Pro

5 - Pentium II

6 - Celeron

7 - Pentium III

PASOS este número se incrementa de acuerdo a los cambios pequeños realizados en el CPU.

MARCA - un nuevo campo de Intel para distinguir alguno de su CPUs. Los valores conocidos son:

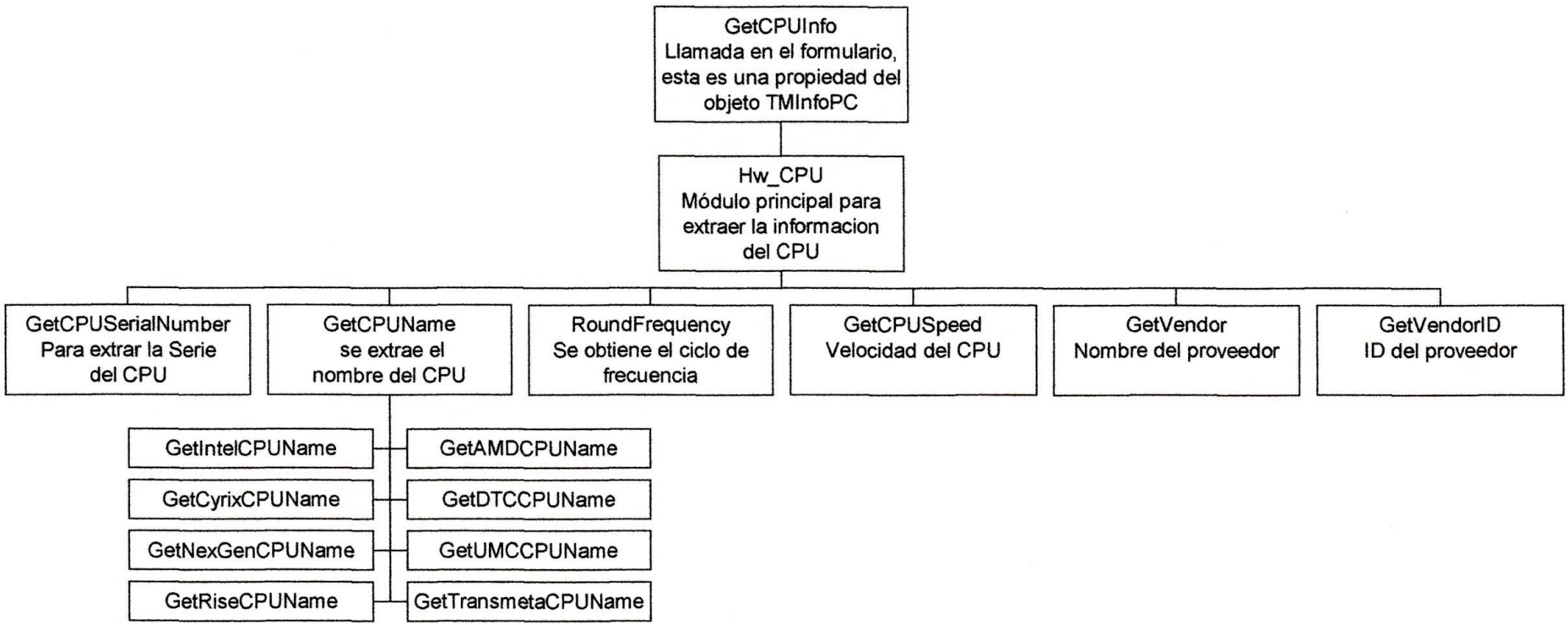
0 – No soportado

2 - Intel Pentium III

3 - Intel Pentium III Xeon

Diagrama de Módulos con funciones y procedimientos utilizadas para extraer la información del procesador

DATOS DEL CPU



En el formulario (Sw_presdatos) de la aplicación se encuentra la llamada al procedimiento GetCPUInfo, el mismo que es una propiedad del objeto TMInfoPC.

El objeto **TMInfoPC**, es una colección de módulos encapsulados en el cual se encuentran diferentes propiedades y una de ellas es la obtención de la información del CPU.

Modulo Hw_CPU:

Es un módulo que nos sirve para extraer toda la información referente al CPU, dentro de este existen varias declaraciones de constantes como son

Const

```
{ CPUID EFLAGS Id bit }          //Comentario

  CPUIDID_BIT =          $200000;

{ CPUID Niveles de Ejecución } //Comentario

  CPUID_MAXLEVEL              : DWORD = $0;
  CPUID_VENDORSIGNATURE : DWORD = $0;
  CPUID_CPUSIGNATURE        : DWORD = $1;
  CPUID_CPUFEATURESET     : DWORD = $1;
  CPUID_CACHETLB              : DWORD = $2;
  CPUID_CPUSERIALNUMBER : DWORD = $3;
  CPUID_MAXLEVELEX          : DWORD = $80000000;
  CPUID_CPUSIGNATUREEX  : DWORD = $80000001;
  CPUID_CPUMARKETNAME1 : DWORD = $80000002;
  CPUID_CPUMARKETNAME2 : DWORD = $80000003;
  CPUID_CPUMARKETNAME3 : DWORD = $80000004;
  CPUID_LEVEL1CACHETLB   : DWORD = $80000005;
  CPUID_LEVEL2CACHETLB   : DWORD = $80000006;
  CPUID_APMFEATURESET    : DWORD = $80000007;
  CPUID_PHYSADDR   : DWORD = $80000008;

{ CPU Proveedor }
  VENDOR_UNKNOWN    = 0;  // Proveedor desconocido
  VENDOR_INTEL          = 1;  // Proveedor Intel
```

```
VENDOR_AMD        = 2;  // Proveedor AMD
VENDOR_CYRIX      = 3;  // Proveedor Cyrix
VENDOR_IDT        = 4;  // Proveedor IDT
VENDOR_NEXGEN     = 5;  // Proveedor NexGen
VENDOR_UMC        = 6;  // Proveedor UMC
VENDOR_RISE       = 7;  // Proveedor Risc
VENDOR_TM         = 8;  // Proveedor TM
```

UNIVERSIDAD TÉCNICA PARTICULAR DE LOJA BIBLIOTECA GENERAL

Estas constantes nos sirven para identificar a los diferentes tipos de proveedores de Procesadores, los cuales tienen su propia identificación internamente en los registros del CPU.

Módulo GetCPUSerialNumber:

Con este módulo obtenemos el numero serial del Procesador, en donde trabajamos directamente con los registros del CPU una vez que se sabe a que tipo de proveedor pertenece con la función CPUID_CPUSIGNATURE, nos ubicamos en los registros mediante código en ensamblador lo cual nos facilita ya que cada proveedor utiliza una instrucción determinada de direcciones dentro del procesador, esto lo hacemos con la función ExecuteCPUID, esta función nos devuelve un valor en Hexadecimal el cual debemos transformarlo a decimal para poderlo mostrar.

Módulo GetCPUName

Aquí se obtiene el fabricante o proveedor del CPU, como se dijo anteriormente, cada proveedor posee su conjunto de instrucciones entonces debemos saber a quien

pertenece para poder trabajar con el mismo, para lo cual se ha creado diferentes tipos de submodulos de este:

- GetIntelCPUName: Para extraer información de procesadores de arquitectura Intel
- GetAMDCPUName: Procesadores de arquitectura AMD
- GetCyrixCPUName: Procesadores de arquitectura Cyrix
- GetDTCCPUName: Procesadores de arquitectura DTC
- GetUMCCPUName: Procesadores Intel como el U486DX2 y el U486sX2
- GetNextGenCPUName: procesadores Intel que salieron en una generación, como son Nx586 y el Nx686
- GetRiseCPUName: Procesadores de arquitectura Rise, estos son por ejemplo:
    - mP6 : iDragon
    - mP6 : iDragon 0.18μm
    - mP6 : iDragon II 0.25μm
    - mP6 : iDragon II 0.18μm
- GetTransMetaCPUName: Procesadores de arquitectura TransMeta. Ejm:
    - Crusoe TM5x00

Módulo RoundFrequency

Módulo o función que nos retorna la frecuencia a la cual trabaja el CPU, esta es distinta a la velocidad del procesador, ya que una CPU puede ser de una velocidad de 550 MHz y trabaja a una velocidad de 100 MHz, por lo que se debe diferenciar estos dos tipos de datos.

Módulo GetCPUSpeed

Retorna la velocidad del Procesador, la cual se mide en Ticks, de acuerdo a la frecuencia con que esta trabajando, la velocidad del CPU esta dentro del conjunto de instrucciones que posee cada tipo de arquitectura del fabricante.

Modulo GetVendor

Mediante este modulo se obtiene una cadena con nombre del fabricante del CPU, la cual esta encapsulada dentro de los registros del procesador.-

Modulo GetVendorID

Cada fabricante de CPU, tiene un único identificador que lo diferencia del resto, por esto es importante este identificador.

```
' Estructura que recupera la configuración de CACHE  de nivel 1

Public Type STRUCT_L1CACHE
  dwSize As Long        'Tamaño, en kb
  dwAssoc As Long       'maneras asociativas
  dwLineSize As Long    'Tamaño de linea
  dwLinesPerTag As Long 'Lineas por etiqueta
  dwPages As Long
End Type
```

La anterior estructura extrae la información de los procesadores que soportan los diferentes niveles de CACHE, por ejemplo el CACHE L1, la información que se obtiene

es el tamaño del CACHE en Kbytes, maneras asociativas, tamaño de línea, línea por etiqueta, que cada CPU actual posee.

```
'Estructura de CACHE L1 TLB

Public Type STRUCT_L1CACHETLB
  dwAssocI As Long    '-way associative for level 1 cache instructions
  dwEntriesI As Long  'number of entries
  dwAssocD As Long    'identically, for data
  dwEntriesD As Long  'the same
  dwPagesI As Long
  dwPagesD As Long
End Type
```

Así como la estructura anterior para el nivel de CACHE L1, existe otro nivel de cache que es el CACHE TLB, por esto los diferentes fabricantes de CPU's cada vez están sacándole más provecho al cache, con un nivel extendido, este es el TLB.

```
' Estructura que obtiene, la familia, modelo y revisiones/avances

Public Type STRUCT_FMS
  dwFamily As Long
  dwModel As Long
  dwStepping As Long
End Type
```

Mediante esta estructura se obtiene la familia, modelo y revisiones/avances de los CPU.

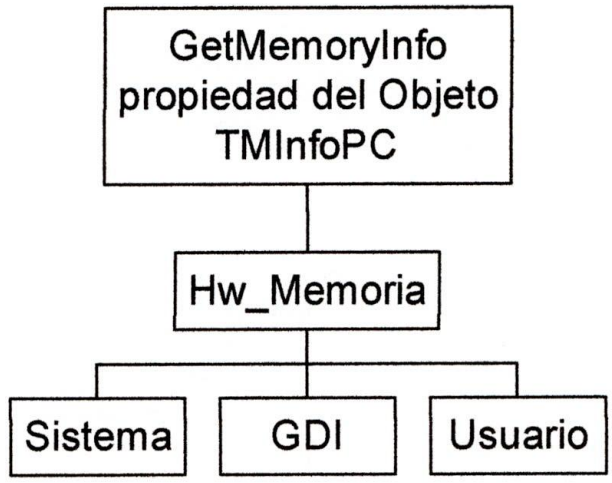
1.2 Determinación de la cantidad de memoria RAM en el computador

RAM significa Memoria de Acceso Aleatorio (Random Access Memory) y es la que se encarga de almacenar la información mientras el computador se encuentra encendido. Esto quiere decir que cuando el computador arranca esta se encuentra vacía

inicialmente, y entonces se lee información del disco duro y se almacena en ella ,el sistema operativo (primero), después, cualquier otra cosa que hagamos. Al trabajar en un procesador de palabras, por ejemplo, la información se almacena aquí. La información solo pasa al disco duro cuando grabamos. Por esto se pierde la información si se apaga el computador sin grabar.

Tener grandes cantidades de memoria RAM es bueno, pero esto no es lo único. La velocidad es importante. Hoy en día se consiguen dos tipos de tarjetas de RAM: SIMM y DIMM (Single in line memory module y Double in line memory module). Los segundos son más nuevos y la mayoría de tarjetas madres nuevas traen de este tipo. Los primeros son el estándar de los computadores menos recientes, pero la basura del mañana. La velocidad de la memoria se mide en nanosegundos (ns=10^(-9) segundos). Mientras menos nanosegundos, mejor es el rendimiento de la memoria, ya que se demora menos en escribir y leer la información.

Diagrama de Módulos para obtener la información de la memoria del PC



GetMemoryInfo, es un procedimiento que se encuentra implementado dentro de la forma Sw_PresDatos en la cual se hace referencia al objeto TMInfoPC, la cual posee la propiedad para extraer datos referentes a la memoria del PC.

Módulo Hw_Memoria

Dentro de este modulo encontramos las propiedades y funciones que hacen referencia a la memoria, como son:

- Memoria Física
- Memoria Física Utilizada
- Memoria Física Disponible
- Memoria Virtual
- Memoria Virtual Utilizada
- Memoria Virtual Disponible

Para obtener la cantidad de memoria existente en nuestro equipo o en cualquier equipo utilizamos funciones del API de windows, en donde se utiliza el archivo **kernell32.dll** dentro del cual podemos realizar la declaración de la siguiente función:

Declare Sub GlobalMemoryStatus Lib "kernel32" (lpBuffer As MEMORYSTATUS)

La función **GlobalMemoryStatus** recupera información sobre la memoria actual disponible. Esta función nos devuelve información sobre la memoria física y virtual. Esta función reemplaza la función de GetFreeSpace, que es una función antigua para versiones anteriores de windows.

El parámetro lpBuffer, es un puntero a una estructura de MEMORYSTATUS en la que se retorna la información sobre disponibilidad de memoria actual. Antes de llamar a esta función, debemos setear la estructura, la estructura es la siguiente:

```
Type MEMORYSTATUS
   dwLength As Long
   dwMemoryLoad As Long
   dwTotalPhys As Long
   dwAvailPhys As Long
   dwTotalPageFile As Long
   dwAvailPageFile As Long
   dwTotalVirtual As Long
   dwAvailVirtual As Long
End Type
```

Esta estructura esta conformada por:

dwlength = longitud o tamaño de la memoria

dwMemoryload = memoria que se ha cargado

dwTotalPhys = memoria física total

dwAvailPhys = memoria física disponible

dwTotalPageFile = memoria total por archivo de pagina

dwAvailPageFile = memoria disponible por archivo de pagina

dwTotalVirtual = memoria virtual total

dwAvailVirtual = memoria virtual disponible.

Para completar esta información debemos declarar una función publica en la cual se realizara los respectivos cálculos, la función posee el siguiente código:

```
Public Sub MemInfo(Used As Long, Physical As Long, Avail As Long, PageFile As
Long, AvailPageFile As Long, Virtual As Long, AvailVirtual As Long)

   Dim mInfo As MEMORYSTATUS

   mInfo.dwLength = Len(mInfo)
   GlobalMemoryStatus mInfo

   Used = mInfo.dwMemoryLoad
   Physical = mInfo.dwTotalPhys / 1024^2
   Avail = mInfo.dwAvailPhys / 1024^2
   PageFile = mInfo.dwTotalPageFile / 1024^2
   AvailPageFile = mInfo.dwAvailPageFile / 1024^2
   Virtual = mInfo.dwTotalVirtual / 1024^2
   AvailVirtual = mInfo.dwAvailVirtual / 1024^2
End Sub
```

En el procedimiento anterior se realiza los respectivos cálculos, ya que los valores retornados por el tipo MEMORYSTATUS es en Bytes, por lo cual aquí lo convertimos a Megabytes, dividiendo el resultado para 1024^2, estos serán los resultados que se los presentara al usuario.

Módulo Sistema

Aquí, se muestra la cantidad de memoria utilizada por el sistema operativo, el cual utiliza varios procesos que se encuentran corriendo en background y que nosotros no podemos verlos, cada uno de estos utiliza recursos lo que hace que disminuya la cantidad de memoria para las aplicaciones de usuario.

Módulo GDI

En este modulo se determina la cantidad de memoria utilizada por la interfaz grafica, la misma que consume una cantidad especifica de la memoria física, y nos disminuye el rendimiento del sistema operativo y además de nuestras aplicaciones que estamos ejecutando.

Módulo Usuario

En este módulo calculamos la memoria física utilizada por el usuario, esto quiere decir los programas y aplicaciones que se esta utilizando o los tenemos abiertos, los cuales nos consumen memoria y reducen el desempeño del sistema operativo.

1.3 Tamaño y Espacio del Disco Duro

La interfaz IDE (más correctamente denominado **ATA**, el estándar de normas en que se basa) es el más usado en PCs normales, debido a que tiene un balance bastante

adecuado entre precio y prestaciones. Los discos duros IDE se distribuyen en canales en los que puede haber un máximo de **2 dispositivos por canal**; en el estándar IDE inicial sólo se disponía de un canal, por lo que el número máximo de dispositivos IDE era 2.

El estándar IDE fue ampliado por la norma ATA-2 en lo que se ha dado en denominar **EIDE** (*Enhanced IDE* o IDE mejorado). Los sistemas EIDE disponen de 2 canales IDE, primario y secundario, con lo que pueden aceptar **hasta 4 dispositivos**, que no tienen porqué ser discos duros mientras cumplan las normas de conectores **ATAPI**; por ejemplo, los CD-ROMs y algunas unidades SuperDisk se presentan con este tipo de conector.

En cada uno de los canales IDE debe haber un dispositivo Maestro (*master*) y otro Esclavo (*slave*). El maestro es el primero de los dos y se sitúa al final del cable, asignándosele generalmente la letra "C" en DOS. El esclavo es el segundo, conectado en el centro del cable entre el maestro y la controladora, la cual muchas veces está integrada en la propia placa base; se le asignaría la letra "D".

Los dispositivos IDE o EIDE como discos duros o CD-ROMs disponen de unos micro interruptores (*jumpers*), situados generalmente en la parte posterior o inferior de los mismos, que permiten seleccionar su carácter de maestro, esclavo o incluso otras posibilidades como maestro con esclavo. Las posiciones de los jumpers vienen indicadas en una pegatina en la superficie del disco, o bien en el manual o serigrafiadas en la placa de circuito del disco duro, con las letras **M** para designar "maestro" y **S** para "esclavo".

Otros avances en velocidad vienen de los **modos de acceso**:

Modo de acceso	Transferencia máxima teórica	Comentario
PIO-0	3,3 MB/s	Discos muy antiguos, de 100 MB o menos
PIO-1	5,2 MB/s	Discos antiguos, de capacidad menor de unos 400 MB
PIO-2	8,3 MB/s	
PIO-3	11,1 MB/s	Discos más o menos modernos, de capacidad superior a unos 400 MB
PIO-4	16,6 MB/s	
DMA-1 multiword	13,3 MB/s	Modos de utilidad dudosa, ya que su velocidad no es mayor que PIO-4
DMA-2 multiword o DMA/16	16,6 MB/s	
UltraDMA o DMA/33	33,3 MB/s	El estándar actual

Aunque en este terreno se barajan las **cifras de transferencia máxima *teóricas***, que no las que físicamente puede alcanzar el disco; los 33,3 MB/s son inalcanzables para cualquier disco duro actual. En realidad, llegar a 10 MB/s con un disco duro UltraDMA es algo bastante difícil de conseguir, las cifras normales están más bien por unos 6 ó 7 MB/s.

Los **modos PIO** se habilitan generalmente mediante la BIOS y dan pocos problemas, aunque en discos duros no actuales a veces la autodetección del modo PIO da un modo un grado superior al que realmente puede soportar con fiabilidad, pasa mucho por ejemplo con discos que se identifican como PIO-4 pero que no son fiables más que a PIO-3.

Los **modos DMA** tienen la ventaja de que liberan al microprocesador de gran parte del trabajo de la transferencia de datos, encargándoselo al chipset de la placa (si es que éste tiene esa capacidad, como ocurre desde los tiempos de los Intel *Tritón*), algo parecido a lo que hace la tecnología SCSI. Sin embargo, la activación de esta característica (conocida como *bus mastering*) requiere utilizar los drivers adecuados y puede dar problemas con el CD-ROM, por lo que en realidad el único modo útil es el UltraDMA.

Se debe tener en cuenta que la activación o no de estas características es opcional y la compatibilidad hacia atrás está garantizada; podemos comprar un disco duro UltraDMA y usarlo en modo PIO-0 sin problemas, sólo estaremos tirando el dinero. Así que si quiere un disco para un 486 que no soporta bus mastering, no se preocupe: compre un disco UltraDMA y seleccione el modo PIO-4, dudo mucho que aprecie la diferencia de rendimiento y la instalación será más sencilla.

Discos duros SCSI

La ventaja de estos discos no está en su mecánica, que puede ser idéntica a la de uno IDE (misma velocidad de rotación, mismo tiempo medio de acceso...) sino en que la transferencia de datos es más constante y casi independiente de la carga de trabajo del microprocesador.

Esto hace que la ventaja de los discos duros SCSI sea apreciable en ordenadores cargados de trabajo, como servidores, ordenadores para CAD o vídeo, o cuando se realiza multitarea de forma intensiva, mientras que si lo único que queremos es cargar Word y hacer una carta la diferencia de rendimiento con un disco UltraDMA puede ser inapreciable.

En los discos SCSI resulta imposible llegar a los 20 MB/s de transferencia teórica del modo Ultra SCSI, y ni de lejos a los 80 MB/s del modo Ultra-2 Wide SCSI, pero sí a cifras quizá alcanzables pero nunca superables por un disco IDE. De lo que no hay duda es que los discos SCSI son **una opción profesional**, de precio y prestaciones elevadas, por lo que los fabricantes siempre escogen este tipo de interfaz para sus discos de mayor capacidad y velocidad.

Verificación de datos del Disco Duro

- La función **Valid**, nos valida para saber si es o no un disco duro o unidad de CD-ROM
- La función **IdeExists** verifica la existencia de dispositivos IDE
- La función **DiskExists**, Se revisa si existen discos en nuestro equipo, y además cuantas unidades hay.
- La función **ATAdevice,** retorna verdadero o falso si son dispositivos ATA
- La función **RemovableDevice** nos retorna verdadero o falso si son o no dispositivos removibles respectivamente
- La función **TotLogCyl** retorna el número total de cilindros lógicos del disco
- La función **TotLogHeads** retorna el número total de cabezas lógicas del disco

- La función **TotLogSPT** retorna el número total de Sectores por Pista o Track lógicos del disco

- La función **CurLogCyl** retorna el número actual de cilindros lógicos del disco

- La función **CurLogHeads** retorna el número actual de cabezas lógicas del disco

- La función **CurLogSPT** retorna el número actual de Sectores por Pista o Track lógicos del disco

- El procedimiento **SerialNumber** nos verifica el numero de serie del disco, este Serial es el que esta impreso en la parte externa, cada disco posee un número único el mismo que viene de fábrica, este es diferente al número d volumen que le asigna el sistema operativa cuando se lo formatea.

- El procedimiento **FirmwareRevision** verifica el número que le asignan también en la fabrica, cada vez que realizan alguna modificación o actualización a los dispositivos, este puede ser igual al de otros, ya que es como versión, los mismos que salen en masa.

- El procedimiento **ModelNumber** extrae el nombre del fabricante y modelo del disco, ya que existen diferentes modelos y además se relacionan con el número de revisión.

1.4 Tarjetas del Computador

Cada equipo posee su propia tarjetas de video, red, sonido, modem ,etc., ya sea que estén integradas en la placa base o individual como un dispositivo adicional, esto depende del MainBoard que nosotros tengamos en nuestro computador.

1.4.1 Tarjeta de Video

Hoy en día todas las tarjetas de vídeo son gráficas e incluyen aceleración por hardware, es decir, tienen "chips" especializados que se encargan de procesar la información recibida desde el bus e interpretarla para generar formas, efectos, texturas, que de otra forma no serían posibles..

En cuanto al tipo de bus, actualmente sólo encontramos dos estándares, el PCI y el AGP. Aunque en un principio el segundo todavía no estaba lo suficientemente bien implementado como para sacarle ventaja al primero

Otro factores a tener en cuenta:

Memoria: En las tarjetas 2D, la cantidad de memoria sólo influye en la resolución y el número de colores que dicha tarjeta es capaz de reproducir. Lo habitual suele ser 1 ó 2 Megas.

Relación entre memoria y Resoluciones máximas

Memoria	Máximas resoluciones y colores			
512 Kb.	1024x768-16 colores	800x600-256 colores		
1 Mb.	1280x1024-16 colores	1024x768-256 colores	800x600-64k colores	640x480-16,7M col.
2 Mb.	1280x1024-256 colores	1024x768-64K colores	800x600-16,7M colores	idem

16 colores = 4 bits.

256 colores = 8 bits.

64k = 65.536 colores = 16 bits

16,7 M = 16.777.216 colores = 24 bits.

En cuanto a la programación en 3D, en un inicio, prácticamente cada fabricante utilizaba su propia API, que es algo así como el "lenguaje" a utilizar para que los programas se comuniquen con el hardware.

De acuerdo a las características que puede tener una tarjeta de video, se ha extraído la información de esta con la función del API de windows, que nos devuelve algunos de estos valores, la función es:

```
Public Declare Function GetDeviceCaps Lib "gdi32" (ByVal hdc As Long, ByVal
                        nIndex As Long) As Long

Public Declare Function GetPrivateProfileString Lib "kernel32" Alias
                        "GetPrivateProfileStringA" (ByVal lpApplicationName As
                        String, ByVal lpKeyName As Any, ByVal lpDefault As
```

String, ByVal lpReturnedString As String, ByVal nSize As Long, ByVal lpFileName As String) As Long

- La función de **GetDeviceCaps** recupera información dispositivo-específica sobre un dispositivo especificado.

  Hdc.- Identifica el contexto del dispositivo.

  Nindex.- Especifica el ítem de retorno. Este parámetro puede ser uno de los valores siguientes:

  DRIVERVERSION, TECHNOLOGY, NUMCOLORS, PHYSICALWIDTH, PHYSICALHEIGHT

- La función de GetPrivateProfileString recupera una cadena de la sección especificada en un archivo de inicialización. Esta función se proporciona para la compatibilidad de aplicaciones de 16-bit basadas en Windows.

Las cuales utilizamos en el código siguiente:

```
Private Sub cb_tvideo_Click()
   Dim X As String, ver, tek
   ver = GetDeviceCaps(hdc, 0)

   List1.AddItem ("")

   Dim d As String, t As Integer, s

   d = String(255, 0)
   t = GetPrivateProfileString("boot.description", "display.drv", "", d, Len(d),
                               "system.ini")
```

```
If t = 0 Then
   s = "Desconocida"
Else
   s = Left(d, t)
End If

List1.AddItem ("Tarjeta de Video:    " & s)

tek = GetDeviceCaps(hdc, 2)
Select Case tek
   Case 0: List1.AddItem ("Tecnologia:   Vektoral Plotter")
   Case 1: List1.AddItem ("Tecnologia:   Raster Display")
   Case 2: List1.AddItem ("Tecnologia:   Raster Printer")
   Case 3: List1.AddItem ("Tecnologia:   Raster Camera")
   Case 4: List1.AddItem ("Tecnologia:   Character Stream, PLP")
   Case 5: List1.AddItem ("Tecnologia:   Metafile, VDM")
   Case 6: List1.AddItem ("Tecnologia:   Display File")
End Select

List1.AddItem ("Version:  " & (ver And &HFF00) / &H100) & "." & (ver And
                        &HFF)
List1.AddItem ("Horizontal:  " & GetDeviceCaps(hdc, 4) & " mm")
List1.AddItem ("Vertical:  " & GetDeviceCaps(hdc, 6) & " mm")
List1.AddItem ("Resolucion:    " & GetDeviceCaps(hdc, 8) & " X " &
                        GetDeviceCaps(hdc, 10))

Dim g
g = GetDeviceCaps(hdc, 14) * 2 ^ GetDeviceCaps(hdc, 12)
Select Case g
   Case "4294967296": List1.AddItem ("Colores:  32 Bit")
   Case "16777216": List1.AddItem ("Colores:  24 Bit")
   Case "65536": List1.AddItem ("Colores:  16 Bit")
   Case "256": List1.AddItem ("Colores:  256")
End Select

End Sub
```

1.4.2 Tarjeta de RED

Cada equipo posee una tarjeta de red para poder conectarse o unirse a una red, la misma que utiliza diferentes parámetros que son constantes las cuales son:

```
Const
        MAX_HOSTNAME_LEN = 132                    // Nombre del Host
```

MAX_DOMAIN_NAME_LEN = 132 // Nombre de Dominio
MAX_SCOPE_ID_LEN = 260
MAX_ADAPTER_NAME_LENGTH = 260 // Nombre de Adaptador
MAX_ADAPTER_ADDRESS_LENGTH = 8 // Dirección del Adaptador
MAX_ADAPTER_DESCRIPTION_LENGTH = 132 // Descripción del adaptador
ERROR_BUFFER_OVERFLOW = 111 // Error de Overflow
MIB_IF_TYPE_ETHERNET = 1 // Tipo de adaptador
MIB_IF_TYPE_TOKENRING = 2
MIB_IF_TYPE_FDDI = 3
MIB_IF_TYPE_PPP = 4
MIB_IF_TYPE_LOOPBACK = 5
MIB_IF_TYPE_SLIP = 6

Estructura para obtener la direccion IP y la mascara de Red del Equipo

DIRECCIONES
Next As Long
IpAddress As String * 16
IpMask As String * 16
Context As Long
Fin Direcciones

Estructura que determina el tipo de adaptador, es decir el nombre de la tarjeta de Red las diferentes características de la misma.

IP_ADAPTER_INFO
Next As Long
ComboIndex As Long
AdapterName As String * MAX_ADAPTER_NAME_LENGTH
Description As String * MAX_ADAPTER_DESCRIPTION_LENGTH
AddressLength As Long
Address(MAX_ADAPTER_ADDRESS_LENGTH - 1) As Byte
Index As Long
Type As Long
DhcpEnabled As Long
CurrentIpAddress As Long
IpAddressList As IP_ADDR_STRING
GatewayList As IP_ADDR_STRING
DhcpServer As IP_ADDR_STRING

```
  HaveWins As Boolean
  PrimaryWinsServer As IP_ADDR_STRING
  SecondaryWinsServer As IP_ADDR_STRING
  LeaseObtained As Long
  LeaseExpires As Long
End
```

Estructura que nos almacena la información permanente de la tarjeta de red.

```
FIXED_INFO
  HostName As String * MAX_HOSTNAME_LEN
  DomainName As String * MAX_DOMAIN_NAME_LEN
  CurrentDnsServer As Long
  DnsServerList As IP_ADDR_STRING
  NodeType As Long
  ScopeId  As String * MAX_SCOPE_ID_LEN
  EnableRouting As Long
  EnableProxy As Long
  EnableDns As Long
End
```

Public Declare Function **GetNetworkParams** Lib "IPHlpApi" (FixedInfo As Any, pOutBufLen As Long) As Long
Public Declare Function **GetAdaptersInfo** Lib "IPHlpApi" (IpAdapterInfo As Any, pOutBufLen As Long) As Long
Public Declare Function **SendMessage** Lib "user32" Alias "SendMessageA" (ByVal hwnd As Long, ByVal wMsg As Long, ByVal wParam As Long, lParam As Any) As Long

Descripción de las funciones

- La función de **GetNetworkParams** recupera parámetros de la red para la computadora local, utiliza el archivo **IPHlpApi,** esta función posee los siguientes parámetros:

  PfixedInfo.- Indicador a una estructura de FIXED_INFO que recibe los parámetros de la red para la computadora local.

PoutBufLen.- Indicador a una variable de ULONG que especifica el tamaño de la estructura de FIXED_INFO. Si este tamaño es insuficiente para almacenar la información, GetNetworkParams rellena esta variable con el tamaño requerido, e ingresa el código de error de ERROR_BUFFER_OVERFLOW.

- La función de **GetAdaptersInfo** recupera información del adaptador para la computadora local.

PadapterInfo.- el Indicador a un buffer que, recibe una lista unida de estructuras de IP_ADAPTER_INFO.

PoutBufLen.- Indicador a una variable de ULONG que especifica el tamaño de la estructura de FIXED_INFO. Si este tamaño es insuficiente para almacenar la información, GetNetworkParams rellena esta variable con el tamaño requerido, e ingresa el código de error de ERROR_BUFFER_OVERFLOW.

1.4.3 Tarjeta de sonido

Se encargan de digitalizar las ondas sonoras introducidas a través del micrófono, o convertir los archivos sonoros almacenados en forma digital en un formato analógico para que puedan ser reproducidos por los altavoces.

Los sonidos que puede percibir el oído humano abarcan las frecuencias de 20 a 20.000 Hz.

La tarjeta de sonido recorre estas ondas tomando muestras del tipo de onda (de su frecuencia), esta operación se realiza con valores variables de muestreo, desde 8.000 hasta 44.100 Hz, a mayor **frecuencia de muestreo** mayor será la calidad de la grabación. Y del **nivel sonoro** de esta onda, esta información se guarda en 8 bits (28 = 256 niveles de sonido) o en 16 bits (216 = 65.536 niveles de sonido). Y en un canal o **Mono** o dos canales o **Estéreo**.

La **calidad telefónica** correspondería a 11.025 Hz, 8bits y Mono. La **calidad de la radio** a 22.050 Hz, 8 bits y Mono, ocupando el archivo el doble que el primero. Y la **calidad del CD** a 44.100 Hz, 16 bits y Estéreo, ocupando el archivo 16 veces más que el primero.

El proceso de reproducción sigue los mismos pasos pero en sentido contrario.

Muchas tarjetas de sonido poseen capacidades **MIDI**; esto significa que en un chip de la tarjeta, sintetizador, se encuentran almacenadas las características de diferentes instrumentos musicales, y la grabación o reproducción de un sonido se hace en referencia a éstos y las notas musicales correspondientes.

Esta información la podemos obtener directamente del registro de windows, en donde se almacenan las características multimedia de cada equipo, las funciones que se ha utilizado son las siguientes:

```
Public Function SoundDesc()
        SoundDesc = GetString(HKEY_LOCAL_MACHINE,
                    "System\CurrentControlSet\Services\Class\MEDIA\0000",
                    "DriverDesc")
```

```
End Function
```

La función anterior nos proporciona la descripción de la tarjeta de sonido existente.

```
Public Function SoundDate()
        SoundDate = GetString(HKEY_LOCAL_MACHINE,
                    "System\CurrentControlSet\Services\Class\MEDIA\0000",
                    "DriverDate")
End Function
```

Esta función nos retorna la fecha del driver que se encuentra actualmente en nuestro computador.

1.4.4 Fax MODEM

Módem es un acrónimo de *MOdulador-DEModulador*; es decir, que es un dispositivo que transforma las señales digitales del ordenador en señal telefónica analógica y viceversa, con lo que permite al ordenador transmitir y recibir información por la línea telefónica.

Los chips que realizan estas funciones están casi tan estandarizados como los de las tarjetas de sonido; muchos fabricantes usan los mismos integrados, por ejemplo de la empresa Rockwell, y sólo se diferencian por los demás elementos electrónicos o la carcasa.

En los MODEMS tenemos que realizar una distinción dependiendo del tipo de bus al que vayan conectados, podemos encontrar modelos para ranura ISA, para PCI o para las mas novedosas AMR. Debido a que el primero está tendiendo a desaparecer, cada vez es más dificil encontrar modelos para él, siendo lo habitual los dispositivos PCI, que además tienen la ventaja del Plug and Play (PnP) que siempre es una ayuda en el momento de su instalación.

Los modelos basados en AMR sólo podremos utilizarlos en las placas más modernas como las que utilizan el chipset i810, y están orientados al mercado de gama baja, debido a que la mayor parte de la funcionalidad del dispositivo está ya implementada en la propia placa base y al igual que ocurre en el caso de los Winmodem su funcionamiento está más basado en el software que en el hardware, lo que repercute en un menor precio de coste pero por el contrario su utilización consume ciclos de CPU y su portabilidad está limitada ya que no todos los sistemas operativos disponen del soporte software adecuado para hacerlos funcionar.

Esta información la podemos obtener directamente del registro de windows, en donde se almacenan las características multimedia de cada equipo, las funciones que se ha utilizado son las siguientes:

```
Public Function ModemDesc()
  ModemDesc = GetString(HKEY_LOCAL_MACHINE, _
                 "System\CurrentControlSet\Services\Class\Modem\0000",
               "DriverDesc")
End Function
```

Función que Retorna la descripción o nombre del MODEM existente

```
Public Function ModemMarca()
  ModemMarca = GetString(HKEY_LOCAL_MACHINE, _
                "System\CurrentControlSet\Services\Class\Modem\0000",
              "ProviderName")
End Function
```

Función que retorna la marca de nuestro MODEM que se encuentra instalado en nuestro equipo.

```
Public Function ModemDate()
  ModemDate = GetString(HKEY_LOCAL_MACHINE, _
                "System\CurrentControlSet\Services\Class\Modem\0000",
              "DriverDate")
End Function
```

Esta función nos retorna la fecha de la versión de los drivers del MODEM que se encuentra funcionando

1.5 Registro de datos Observados

De acuerdo a los capítulos anteriores, se ha determinado, que cada uno de los dispositivos que posee un equipo tiene un registro individual por el cual se los puede identificar, el Sistema Operativo contiene algunos archivos que por medio de parámetros se extrae esta información.

Según lo que hemos observado, los archivos DLL que tiene el sistema operativo recogen muchos datos, los cuales no deben faltar en un computador cuando se esta ejecutando algún comando, los archivos mas importantes que se ha utilizado son:

- **getcpu.dll.-** mediante este archivo se obtiene los datos el CPU invocando a la función: *GetProcessor*

- **kernell32.dll.-** este archivo posee una gran cantidad e funciones internas, por lo que debemos especificar que es lo que deseamos obtener, a continuación se muestra las funciones que se ha utilizado de este archivo:

    - *GlobalMemoryStatus*: nos retorna la cantidad de memoria tanto física como virtual, esta función utiliza una estructura de nombre MEMORYSTATUS, en la cual se obtiene toda esta información.

- **DSIdeInf.DLL.-** Esta librería no se encuentra en el sistema operativo y nos proporciona los datos del Disco Duro, por lo que ha sido necesario implementarla en código de bajo nivel como ensamblador y además utilizando el lenguaje Delphi, de acuerdo a los parámetros podemos obtener el numero de serie real del dispositivo (disco duro), como son cilindros, cabezas, pistas y sectores, así como también el tipo de dispositivo que tenemos como puede ser fijo o removible con lo cual nosotros tenemos datos exactos, en caso de existir alguna duda sobre nuestro disco(s).

- **IPHIpApi.-** esta es una librería que se encuentra en el sistema operativo pero que no se encuentra como archivo como los anteriores pero que puede ser invocado el cualquier lugar, mediante esta librería nosotros podemos obtener los datos de la tarjeta de RED si llamamos a las siguientes funciones:

    - *GetNetworkParams*, esta función nos retorna los parámetros de la red actual.
    - *GetAdaptersInfo*, retorna información del adaptador existente

- **gdi32.dll,** asi mismo esta librería nos retorna diferentes parámetros que sirven para poder recuperar la información de todos los dispositivos existes.

- **advapi32.dll.-** esta librería posee algunas funciones las mismas que se las utiliza para poder manipular el registro de windows, las funciones, mas utilizadas son las siguientes:

    - *RegCloseKey*
    - *RegCreateKey*
    - *RegDeleteKey*
    - *RegDeleteValue*
    - *RegOpenKey*
    - *RegQueryValueEx*
    - *RegSetValueEx*

Cada una de estas funciones manipulan el registro del sistema Operativo, como son creando, cerrando, borrando claves, las mismas que tienen y deben ser tratadas con especial cuidad ya que mediante estas claves se basa el funcionamiento del S.O.

Estas vienen a ser las principales librerías que utiliza el sistema operativo para mantenerse en funcionamiento, si alguna de estas llega a dañarse, eliminarse o moverse a otro lugar que no sea el directorio o carpeta c:\windows\system nuestro S.O. llegara a colapsar y se tendrá que remplazar o en su defecto volver a instalar el sistema.

CAPITULO II

2. MONITOREO DE LOS RECURSOS SOFTWARE

2.1. Procesos en Ejecución

En primer lugar Qué es un proceso?

"Un proceso es una aplicación ejecutándose que consiste en un espacio de direcciones virtual y privado, código, datos y recursos del sistema operativo, tales como archivos, permisos y objetos para sincronización, que son visibles para el proceso"[1]

Como se desprende de tal definición, un proceso posee varios objetos: el espacio de direcciones, el/los módulo/s en ejecución y cualquier cosa que estos módulos creen o abran. Como mínimo, un proceso debe consistir de un módulo de ejecución, un espacio de direcciones privado y un thread[2]. Cada proceso debe poseer al menos un thread. Ni bien Windows crea un proceso, le asigna un sólo thread, el que generalmente comienza su ejecución a partir de la primera instrucción que hay en el módulo. Si el proceso más tarde necesita más threads, las puede crear.

Cuando Windows recibe un comando para crear un proceso, también crea un espacio de direcciones de memoria privado para el proceso y luego mapea el módulo ejecutable dentro de ese espacio. Justo después que arranca la ejecución del thread primario del proceso.

En Win32, Cada proceso en ejecución se crea llamando a la función CreateProcess, que tiene la siguiente estructura:

---

[1] Manual de referencia de Microsoft API Win32
[2] Fila de tareas listas para ejecutarse

CreateProcess proto

```
lpApplicationName:DWORD,\
lpCommandLine:DWORD,\
lpProcessAttributes:DWORD,\
lpThreadAttributes:DWORD,\
bInheritHandles:DWORD,\
dwCreationFlags:DWORD,\
lpEnvironment:DWORD,\
lpCurrentDirectory:DWORD,\
lpStartupInfo:DWORD,\
lpProcessInformation:DWORD
```

A continuación se describen cada uno de los parámetros de esta estructura:

lpApplicationName	El nombre de un archivo ejecutable que se quiere correr, con o sin el path. Si este parámetro es nulo, el nombre del archivo ejecutable deberá ser provisto en el parámetro lpCommandLine.
lpCommandLine	Argumentos de la línea de comando para el programa a ejecutar. Nótese que si lpApplicationName es nulo, este parámetro debe incluir además el nombre del archivo a ejecutar. Ejemplo: "notepad.exe readme.txt".
lpProcessAttributes y lpthreadAttributes	Especifican los atributos de seguridad para el proceso y el thread primario. Si se deja nulo, adquieren los atributos por default.
bInheritHandles	Una flag que especifica si el nuevo proceso debe heredar todos los handles abiertos por el proceso parent.
dwCreationFlags	Varias flags que determinan el comportamiento del

	proceso a crear, tales como si se quiere que el nuevo proceso sea suspendido apenas creado (para examinarlo o modificarlo antes que comience su ejecución). Se puede además especificar la clase de prioridad del thread del nuevo proceso, que se usa para determinar la prioridad de ejecución. Normalmente se usa la flag NORMAL_PRIORITY_CLASS.
lpEnvironment	Un puntero al block de variables de entorno que contiene las string de entorno para el nuevo proceso. Si este parámetro se deja en NULL, el nuevo proceso hereda el block de entorno de su proceso parent.
lpCurrentDirectory	Un puntero al string que especifica el drive y directorio actual del proceso a crear. Se deja en NULL si se desea que lo herede del proceso parent.
lpStartupInfo	Puntero a una estructura STARTUPINFO que especifica cómo se debe ver la ventana principal del nuevo proceso. Si no se pretende nada muy especial, la estructura STARTUPINFO puede llenarse con los valores de la que utiliza el proceso parent, que se averigua llamando la función GetStartUpInfo.
lpProcessInformation	Apunta a una estructura PROCESS_INFORMATION que recibe información de identificación sobre el nuevo

	proceso. Esta estructura tiene los siguientes parámetros: PROCESS_INFORMATION STRUCT hProcess HANDLE ? ; handle del proceso child hThread HANDLE ? ; handle al thread primario del proceso child dwProcessId DWORD ? ; ID del proceso child dwThreadId DWORD ? ; ID del thread primario del proceso child PROCESS_INFORMATION ENDS

Nótese que un handle de proceso es un objeto diferente al ID del proceso, ya que este último es un identificador único del proceso en el sistema, mientras que el handle es un valor retornado por Windows para usarlo en otras funciones API en las que debe hacerse referencia al proceso. Un handle de proceso no puede usarse para identificar un proceso porque no es único.

Durante la llamada a la función CreateProcess, se crea un nuevo proceso y la API retorna de inmediato. Se puede verificar si el nuevo proceso está todavía activo llamando a la función

GetExitCodeProcess con la siguiente sintaxis:

```
GetExitCodeProcess proto
        hProcess:DWORD,
        lpExitCode:DWORD
```

Si en este llamado no se produce error, lpExitCode contiene la información de status generada al finalizarse el proceso en cuestión. Si el valor en lpExitCode es igual a STILL_ACTIVE, entonces es que el proceso todavía está corriendo.

Se introduce en este momento el concepto de **hilo** de ejecución, como un concepto un tanto diferente del concepto de proceso, aunque ciertamente relacionados entre sí. Un proceso es una unidad a la cual se le pueden asignar recursos. De esta forma, vemos que a un proceso se le asigna espacio en memoria, tanto virtual como real; archivos; dispositivos y canales de I/O; etc. Un hilo es una unidad a la cual se le asigna el procesador. Por lo tanto, es una unidad que está en uno de los estados ("Ready", en ejecución, etc.), con su propio contexto, y "stack", y con acceso a los recursos del proceso al que pertenece.

La ventaja fundamental del uso de hilos es que se aumenta el rendimiento del sistema, es decir, el sistema es más eficiente con el uso de hilos. En un sistema que no maneja hilos, las tareas a que podrían realizarse por varios hilos de un solo proceso tendrían que llevarse a cabo por varios procesos. Es mucho más fácil y rápido crear y destruir hilos que procesos. Además, la comunicación entre hilos es más fácil que entre procesos, pues de este última forma debe intervenir el sistema operativo para cotejar permisos de acceso, mientras que en la comunicación entre hilos de un proceso el sistema operativo no tiene que intervenir pues todos los hilos de un proceso comparten y tienen acceso a los mismos recursos.

Las operaciones que se asocian con hilos y que hacen que un hilo cambie de estado son:

"spawn": creación - de un proceso, lo que crea un hilo, creación de un hilo de un proceso que ya existe, por otro hilo de ese proceso

bloquear: esperar por que un evento ocurra

"unblock": cuando el evento por el que espera el hilo ocurre, y el hilo pasa a "Ready"

fin: fin de la ejecución del hilo

Suspender un hilo no tiene sentido. Suspender es una acción que el sistema lleva a cabo para reducir su carga de trabajo. Por lo tanto, es el proceso completo el que se suspende con todos sus hilos.

Existen dos filosofías para implantar la administración de hilos: a nivel de kernel o a nivel de aplicación.

A nivel de aplicación significa que el kernel del sistema operativo no reconoce la existencia de los hilos, si no que es la aplicación quien los administra.

Las ventajas de esta filosofía son:

1. Mayor eficiencia en la ejecución de la aplicación pues no hay que cambiar a modo kernel ni ejecutar el sistema operativo cuando se cambia de un hilo a otro;
2. La aplicación puede manejarse en hilos no importa el sistema operativo bajo la que corra;

3. La forma en que se asigna el procesador a los hilos puede variar de aplicación en aplicación, y cada aplicación puede escoger el algoritmo de asignación de procesador que más le convenga.

Las desventajas de esta filosofía son:

1. En un sistema con múltiples procesadores los hilos se ejecutan en un solo procesador, pues el sistema operativo asigna un procesador a cada proceso y no reconoce la existencia de los hilos;

2. Si un hilo tiene que esperar por un evento y se bloquea, todos los hilos se bloquean a la vez pues el sistema operativo bloquea al proceso.

Los hilos se administran a nivel de kernel cuando toda la administración de ellos recae en el sistema operativo. La ventaja de esta filosofía, además de que se superan las desventajas de la filosofía anterior, son: (3) las tareas del sistema operativo se pueden manejar como hilos.

Algunos sistemas operativos trabajan combinando ambas filosofías.

2.2. Recursos Utilizados por las Aplicaciones

En este punto todas las aplicaciones utilizan tres archivos los cuales consumen recursos del sistema operativo y disminuyen su rendimiento, cada uno con sus características especiales. Estos son:

* KERNEL386.EXE el cual es el responsable de controlar y asignar todos los recursos de manejo de memoria, carga de aplicaciones y la planificación y ejecución de tareas.

* USER.EXE crea y mantiene las ventanas en la pantalla. Trata todas las peticiones de mover, crear, tamaño, o cerrar una ventana. También es el responsable del manejo de los iconos y otros componentes gráficos. Además es el responsable de la entrada a las aplicaciones desde teclado, ratón o cualquier dispositivo de entrada.

* GDI.EXE es el "Graphics Device Interface". Es el responsable de ejecutar operaciones graficas para crear imágenes en la pantalla (y otros dispositivos).

Los componentes, USER, GDI y KERNEL que ejecuta el Sistema Operativo son de 16 bites. Además de estos 3, windows 95 / 98, carga el USER32, GDI32, y KRNL32 que son los 3 respectivos componentes para tareas de 32 bits.

El API es el conjunto de funciones que puede utilizar cualquier tarea windows para realizar una acción. Por ejemplo "crear una ventana". Evidentemente hay dos API's una de 16 y otra de 32 (cada tarea, dependiendo del tipo -16-32-, llama a la correspondiente función).

El API del GDI32 tiene los siguientes subcomponentes:

- GDI32 como tal
- Adaptador tipográfico TrueType

- Archivos TrueType (usa archivos proyectados en memoria).
- Perfiles de color (usa archivos proyectados en memoria).
- Correspondencia de colores de imágenes (ICM)
- INTERFAZ de salto al subsistema de 16 bites.

Por tanto una tarea de 32 que utilice el GDI32, siempre salta al subsistema de 16, que a su vez contiene (API de 16):

- GDI16
- Minicontrolador de visualización
- Minicontrolador de Impresora
- Controlador universal de impresora
- Motor DIB grafico
- Visualización o Impresora (drivers ya específicos del fabricante).

Todas las llamadas a funciones de esta API, que vayan creando "objetos" (asignación de memoria, iconos, ventanas, etc...), consumen una pequeña cantidad de "recursos del SO".

Contra el consumo de estos recursos, no hay solución. Además, lo "grave" del tema, es que es responsabilidad del programa que "consume" recursos, el liberarlos. Por tanto un programa mal codificado (mal diseñado), o que termina anormalmente, nos dejará recursos "gastados" en la pila, que no se recuperarán hasta que reiniciemos windows.

2.3. Liberación de Recursos

Los componentes de la computadora se comunican con el procesador a través de recursos del sistema. Estos recursos del sistema incluyen solicitudes de interrupción, (IRQs), accesos de memoria directa (DMAs) y direcciones de entrada/salida (E/S).

- Las IRQs consisten de mensajes que los componentes de la computadora envían al procesador. Cuando un dispositivo necesita que el procesador realice una operación, éste envía un mensaje para "interrumpir" la actividad del procesador.
- DMAs son líneas directas a la memoria a través de las que un componente puede obtener acceso a la RAM de la computadora.
- Cada dispositivo de entrada o salida debe contar con su propia dirección de E/S para que el procesador pueda controlar las ubicaciones de las que recibe entradas y a las que envía salida.

El procesador puede manejar únicamente un número limitado de cada tipo de recurso de sistema y, si dos componentes intentan usar el mismo recurso, estos componentes no funcionarán de manera adecuada. Esto puede ocurrir si instala un dispositivo de manera incorrecta en la computadora.

Por lo general, no es necesario preocuparse en cuanto a los recursos del sistema porque la característica Conectar y Usar (Plug and Play) en Windows los maneja por usted. Sin embargo, si instala un dispositivo sin esta característica y un programa de instalación le indica entrar un recurso de sistema o, si tiene problemas con un dispositivo que ha instalado, puede hacer que le examinen los recursos del sistema.

En esta sección debemos finalizar (matar) todos los hilos de la aplicación que queremos que deje de funcionar, aquí se utiliza las funciones predeterminadas del Sistema Operativo.

Dispose()

FreeMem()

Dispose(), esta función libera todos los thread (hilos) que esta utilizando en ese momento la aplicación, por lo que el Sistema Operativo le quita el control al mismo.

FreeMem().- Mediante esta función se libera la memoria utilizada por los thread (hilos) que estaba utilizando la aplicación.

El proceso puede ser terminado a la fuerza llamando la función TerminateProcess, con la sintaxis:

TerminateProcess proto hProcess:DWORD, uExitCode:DWORD

Es posible especificar el valor que se desee para el código de salida del proceso. TerminateProcess no es una manera limpia de terminar un proceso ya que cualquier .DLL vinculado a él no será notificado sobre la finalización del proceso.

CAPITULO III

3. VERIFICACIÓN DE SOFTWARE INSTALADO

3.1. Registro del Sistema Operativo

El registro, presenta de forma jerarquizada toda la información que windows y el resto de las aplicaciones escriben en él.

El registro principal (visible con "regedit"), consta de 6 claves principales. Cada clave puede contener datos, llamados "valores" y a su vez puede contener subclaves adicionales. Su visión se asemeja a la visión de un disco desde el explorador, con archivos (valores) y carpetas (subclaves), que a su vez pueden contener mas archivos y mas carpetas.

Claves generales:

HKEY_LOCAL_MACHINE: Información de los componentes hardware instalados. Configuración del software de la maquina. Esta clave es "común" para todos los usuarios de la maquina.

HKEY_CURRENT_CONFIG: Apunta a una rama de HKEY_LOCAL_MACHINE\CONFIG que contiene información sobre la configuración "actual" del hardware.

HKEY_DYN_DATA: Apunta a una rama de HKEY_LOCAL_MACHINE que contiene la información dinámica del estado de los dispositivos. así como toda la parte de PnP. Esta información es cambiante según se añaden o quitan dispositivos al sistema.

Además incluye información sobre el estado actual de los dispositivos, así como de los posibles problemas.

UNIVERSIDAD TÉCNICA PARTICULAR DE LOJA BIBLIOTECA GENERAL

HKEY_CLASSES_ROOT: Apunta a la rama de HKEY_LOCAL_MACHINE que contiene ciertas configuraciones de software. información del OLE y asignación de operaciones de arrastrar y soltar, y ciertas abreviaturas que de echo son vínculos OLE. Igualmente contiene la descripción del interfaz de usuario.

HKEY_USERS: Contiene la información de TODOS los usuarios que se han conectado a la maquina. Incluye tanto la información genérica (.DEFAULT) como la especifica de cada usuario. La configuración general está disponible para todos los usuarios. Esta clave, contiene subclaves con el nombre de cada usuario y su perfil que se ha conectado a la maquina.

HKEY_CURRENT_USER: Esta clave apunta a la rama HKEY_USERS para el usuario que está en ese momento conectado

Descripción Detallada de las Claves mas Importantes.

HKEY_LOCAL_MACHINE

CONFIG : contiene la información de las distintas configuraciones hardware de la maquina. Puede tener las configuración que tiene el PC, cuando está conectado a una red, cuando no lo está, cuando quitamos una tarjeta,....etc. Cada configuración posible tiene asignado un identificador único Este identificador de la configuración tiene una

subclave bajo la clave "config". Cada configuración, aparece en la lista de "perfiles" de hardware en la opción Sistema del Panel de Control.

En la mayoría de las situaciones, el identificador de configuración está asignado a una configuración única. Normalmente casi nadie tiene varios "perfiles" de hardware definidos y arranca con uno u otro. Lo normal para los usuarios domésticos como nosotros es una única configuración. Pero no debe perderse de vista que es posible una configuración "multiperfil" y que windows es capaz en la arrancada de solicitarnos que perfil de hardware queremos.

ENUM : Los "enumeradores" de bus de windows son los que aquí construyen el árbol del hardware. Se asigna un código único de identificación para cada dispositivo.

Las subclaves contiene información sobre el tipo de dispositivo, la letra asignada a la unidad, por ejemplo. El fabricante y además la información de la los controladores de los componente de la RED.

Típicamente, dentro de esta clave se enumeran:

ESDI : Dispositivos de disco Fijo

FLOP : Dispositivo de disco flexible.

ISAPNP : Dispositivo PnP ISA

MONITOR : Monitores

NETWORK : Protocolos, servicios y vínculos de red.

ROOT : Dispositivos legados (no PnP)

Evidentemente si miramos en nuestra clave de registro, nos encontraremos con otras subclaves que contiene al menos las anteriores, más las características genéricas de nuestra maquina.

SOFTWARE : Contiene la información de configuración sobre el software instalado. Estas entradas se aplican a "cualquier" usuario conectado a esta maquina. Contiene las definiciones para asociación de archivos e información OLE.

Esta rama, contiene varias subclaves, incluyendo la subclave "Classes", además de las subclave "description" para todo el software.

Subclaves principales:

CLASSES: Describen todo tipo de documentos y dan información sobre OLE y asociaciones de extensiones a archivos que pueden utilizar aplicaciones.

HKEY_CLASSES_ROOT no es nada más que una "alias" de esta subclase. Aquí están contenidas dos tipos de subclaves.

- Subclases extensiones de archivos, que especifican la definición de clase asociada con el archivo que tiene la extensión seleccionada.

- Claves de definición de clase, en donde se especifican las propiedades OLE y de la interfaz de ordenes de una clase de documento. Si una aplicación permite DDE (intercambio dinámico de datos), la subclase "Shell" puede contener subclaves "Open" y "Print" que definen las ordenes DDE.

DESCRIPTION: Esta subclave contiene los nombres y números de versión del software instalado.

La información especifica de usuario sobre la configuración se almacena en el camino lógico relativo bajo HKEY_CURRENT_USER.

La información en cada subclave la añade cada nueva aplicación al instalarse.

Durante la instalación, las aplicaciones graban con el siguiente formato:

HKEY_LOCAL_MACHIME\SOFTWARE\Nombre_de_compañia\Producto\Version

Evidentemente, el propio Microsoft, graba aquí una clave llamada:

HKEY_LOCAL_MACHINE\SOFTWARE\MICROSOFT

y dentro de ella una subclave llamada WINDOWS\CurrentVersion la cual es importantísima ya que es el "núcleo" de windows.

La subclave SETUP que cuelga de la anterior, es usada internamente por windows durante su arranque.

SYSTEM : Contiene los parámetros completos para los controladores de dispositivos y servicios que son cargados en windows. Todos los datos que controlan el arranque de Windows están aquí. Este conjunto tiene a su vez dos partes:

CONTROL

SERVICES

Vamos a ver un poco mas con detalle estas ultimas ramas.

CONTROL: Contiene parámetros de arranque del sistema, incluyendo las configuraciones de arranque y apagado, rendimientos, esquemas de teclado, idiomas, etc.... Subclaves típicas dentro de el son:

ComputerName; Nombre del PC

FileSystem: Tipo y configuración sistema de archivos

IDConfigDB: Identificación de la configuración actual

Keyboard layouts: Lista de DLLs para el idioma del teclado

Media Resources: información de los componente multimedia

Network Provider: descripción de los proveedores de RED

NLs : información sobre los lenguajes nacionales permitidos

PerfStats: Estadísticas que se ven en "monitor del sistema"

Print: Impresoras actuales y sus entornos de impresión

Session Manager: Variables globales y lista de programas

TimeZonInformation: configuración de la Zona Horaria

Update: Tipo de instalación

VMM32: Los nombres de las VXD que se han combinado para generar el controlador de dispositivos virtuales VMM32.VXD (esto depende de cada maquina, y se "combina" o genera al instalar windows).

SERVICES: Presenta la lista de controladores del núcleo, controladores de sistemas de archivos y controladores de los servicios que pueden ser cargados al iniciar windows. Subclaves típicas dentro de el, son:

Agent_name: Subclaves para cada agente del sistema instalado (Monitor de Red, SNMP, etc....)

Arbitrators: Arbitros para administrar recursos entre dispositivos competidores (árbitros de direcciones, DMA, E/S e IRQ)

Class: Definidos aqui "todas" las clases de dispositivos que permite el sistema operativo (discos, monitores, ratones....)

MSNP32, MWNP32: Subclaves para la RED en modo protegido de 32 bit, incluye información sobre el acceso y los proveedores de seguridad.

VxD: Controladores virtuales (VxD) que se cargan en el inicio de windows. Independientemente de estos se carga SIEMPRE todas las VxD que se encuentran presentes en el directorio IOSUBSYS.

Con esto se finaliza la clave HKEY_LOCAL_MACHINE.

Es importante volver a resaltar que las claves:

HKEY_CURRENT_CONFIG

HKEY_CLASSES_ROOT

HKEY_DYN_DATA

que nos muestra el programa regedit, no son mas que "alias" de otras subclaves del mismo nombre de HKEY_LOCAL_MACHINE, por lo que modificando cualquiera de ellas, (el alias o la original), se modifican ambas.

Esto muchas veces es causa de confusión, ya que vemos información que aparentemente está duplicada.

La otra (única) clave real además de HKEY_LOCAL_MACHINE, es HKEY_USERS ya que realmente la clave HKEY_CURRENT_USER es un alias que apunta en ese momento al usuario actual que está contenido en HKEY_USERS.

HKEY_USERS contiene la subclave .Default además de todos los perfiles de usuario previamente cargados por los usuarios que han accedido. La información de .Default, se usa para crear un perfil de usuario que ha accedido sin perfil de usuario personal.

La subclave .Default contiene claves para AppEvents, Panel de Control, teclados, red, y software, etc.

Esta es la estructura de la base de registro. Evidentemente está accesible tanto a windows como al resto de programas. El problema surge cuando algún programa no es "respetuoso" y empieza a machacar claves de otros.

Hay que recordar que el registro es la base del funcionamiento de windows. Sin el registro windows es incapaz de funcionar y con el registro dañado, nos podremos encontrar cualquier cosa. Por ello, se aconseja no utilizar NINGUNA utilidad para mantener el registro.

3.2. Detección de Programas Instalados

Antes de realizar una descripción sobre los programas instalados en un PC también debemos saber que el Sistema Operativo es la base principal para tener en funcionamiento nuestro equipo lo que hace que sea uno de los programas que sea el primero en ser instalado .

Sistema operativo, software básico que controla una computadora. El sistema operativo tiene tres grandes funciones: coordina y manipula el hardware de la computadora, como la memoria, las impresoras, las unidades de disco, el teclado o el mouse; organiza los archivos en diversos dispositivos de almacenamiento, como discos flexibles, discos duros, discos compactos o cintas magnéticas, y gestiona los errores de hardware y la pérdida de datos.

Funciones y características de los sistemas operativos.

Funciones de los sistemas operativos.

- Aceptar todos los trabajos y conservarlos hasta su finalización.

- Interpretación de comandos: Interpreta los comandos que permiten al usuario comunicarse con el ordenador.

- Control de recursos: Coordina y manipula el hardware de la computadora, como la memoria, las impresoras, las unidades de disco, el teclado o el Mouse.

- Manejo de dispositivos de E/S: Organiza los archivos en diversos dispositivos de almacenamiento, como discos flexibles, discos duros, discos compactos o cintas magnéticas.

- Manejo de errores: Gestiona los errores de hardware y la pérdida de datos.

- Secuencia de tareas: El sistema operativo debe administrar la manera en que se reparten los procesos. Definir el orden. (Quien va primero y quien después).

- Protección: Evitar que las acciones de un usuario afecten el trabajo que esta realizando otro usuario.

- Multiacceso: Un usuario se puede conectar a otra máquina sin tener que estar cerca de ella.

- Contabilidad de recursos: establece el costo que se le cobra a un usuario por utilizar determinados recursos.

Características de los sistemas operativos

En general, se puede decir que un Sistema Operativo tiene las siguientes características:

- Conveniencia. Un Sistema Operativo hace más conveniente el uso de una computadora.

- Eficiencia. Un Sistema Operativo permite que los recursos de la computadora se usen de la manera más eficiente posible.

- Habilidad para evolucionar. Un Sistema Operativo deberá construirse de manera que permita el desarrollo, prueba o introducción efectiva de nuevas funciones del sistema sin interferir con el servicio.

- Encargado de administrar el hardware. El Sistema Operativo se encarga de manejar de una mejor manera los recursos de la computadora en cuanto a hardware se refiere, esto es, asignar a cada proceso una parte del procesador para poder compartir los recursos.

- Relacionar dispositivos (gestionar a través del kernel). El Sistema Operativo se debe encargar de comunicar a los dispositivos periféricos, cuando el usuario así lo requiera.

- Organizar datos para acceso rápido y seguro.

- Manejar las comunicaciones en red. El Sistema Operativo permite al usuario manejar con alta facilidad todo lo referente a la instalación y uso de las redes de computadoras.

- Procesamiento por bytes de flujo a través del bus de datos.

- Facilitar las entradas y salidas. Un Sistema Operativo debe hacerle fácil al usuario el acceso y manejo de los dispositivos de Entrada/Salida de la computadora.

Con la aclaración anterior sobre el sistema operativo nos adentramos en el Sistema Operativo Windows que es nuestro caso de estudio, para lo cual realizamos una análisis del registro del sistema lo que nos permitirá obtener lo que nosotros deseamos.

Dentro de windows existe lo que se llaman las claves y valores que se encuentran en el registro, estas están clasificadas en diferentes opciones, en una de estas encontramos que se encuentran todos los programas que se encuentran instalados en el equipo.

Durante la instalación, las aplicaciones graban con el siguiente formato:

HKEY_LOCAL_MACHIME\SOFTWARE\Nombre_de_compañia\Producto\Version

Evidentemente, el propio Microsoft, graba aquí una clave llamada:

HKEY_LOCAL_MACHINE\SOFTWARE\MICROSOFT

HKEY_LOCAL_MACHINE
Software
Microsoft
Windows
CurrentVersion
uninstall\

Dentro de esto esta una subclave que se denomina DESCRIPTION: Esta subclave contiene los nombres y números de versión del software instalado. La información especifica de usuario sobre la configuración se almacena en el camino lógico relativo bajo HKEY_CURRENT_USER.

La información en cada subclave la añade cada nueva aplicación al instalarse.

Aquí nosotros encontramos un listado de carpetas que dentro de cada una existe una clave que se denomina **DisplayName**, y este es el nombre con el que se muestra cualquier programa una vez que se lo ha instalado.

Al realizar nosotros una función que nos recorra todas estas carpetas y que nos busque la clave DisplayName entonces obtenemos el nombre de cada aplicación.

3.3. Verificación de Características de los Programas

Toda aplicación Windows, independientemente de la versión del sistema en que se haya desarrollado, puede tener unos ciertos requerimientos para poder ejecutarse, actualmente existen diferentes versiones de Windows, desde el ya antiguo Windows 95 hasta el más novedoso Windows 2000, pasando por Windows 98 y Windows 98 SE así como Windows NT y sus varios *Service Pack*, en el futuro habrá disponibles nuevas versiones de Windows 2000.

Lo anterior significa que nuestra aplicación podría, en determinados casos, estar ejecutándose en versiones del sistema diferentes de las que se habían provisto en un

momento. Si no tenemos esto en cuenta dejamos al azar los problemas que pudieran surgir y que pueden ir desde la imposibilidad total de ejecutar el programa, en los casos más extremos, hasta problemas relativamente menores como errores en la localización de carpetas del sistema o usuario, incorrecto posicionamiento de las ventanas en pantalla, etc.

Una aplicación no debe dar nada por supuesto y ha de adaptarse al entorno del usuario, por ejemplo obteniendo la posición de las carpetas del sistema, usando cuadros de diálogo estándar, almacenando información del usuario y comprobando la plataforma de ejecución.

Uno de los problemas que plantea el desarrollo de aplicaciones para Windows, centrándonos en Win32 y olvidando versiones previas como Windows 3.1, consiste en determinar las funciones disponibles en el propio sistema. Éstas cambian de una *versión* a otra, de tal forma que *algo que es* completamente posible en Windows 98 no lo es en NT 4 o viceversa. El problema se agrava porque, además, no hay que tomar en consideración sólo la versión del sistema operativo sino también la de otros elementos como la interfaz del sistema.

Determinar qué versión del sistema operativo está ejecutándose es relativamente fácil, pero en ocasiones necesitaremos ir más allá para saber si una determinada capacidad está o no disponible.

En ocasiones una aplicación puede tener dependencias no sólo del software, las posibilidades de la versión de Windows en que se esté ejecutando, sino incluso del propio sistema físico, al precisar una determinada cantidad de memoria o un cierto tipo de procesador. También puede resultar interesante, sobre todo al trabajar en entornos

de red, conocer el nombre del ordenador y el usuario que lo está utilizando.

Cada aplicación posee sus propias características, las mismas que se las determina al momento de realizar la compilación, y además depende del lenguaje de programación en que se hayan hecho.

Lo que podemos mostrar es su fecha de creación y cual es su fabricante.

CAPITULO IV

4. ANÁLISIS DEL SISTEMA

1. ***PANORAMA GENERAL***: Herramienta de Software que permita realizar el monitoreo de los recursos Hardware y Software de la Red Windows de la UPSI que pertenece a la UTPL.

2. ***BENEFICIARIO***: Unidad de Proyectos y Sistemas Informáticos de la Universidad Técnica Particular de Loja

3. ***METAS:***

- ✓ Testear todo el Hardware interno de un Computador.
- ✓ Revisar el Software actualmente instalado en un Computador.
- ✓ Observar los procesos en ejecución.

4. ***FUNCIONES***

Ref.	**FUNCIONES**	***CATEGORÍA***
1.1	Permitir Visualizar datos del Hardware interno de un Computador	Evidente
1.2	Permitir Seleccionar un dispositivo especifico de hardware para observar sus características	Evidente
1.3	Permitir visualizar el Software instalado actualmente en un Computador	Evidente
1.4	Permitir ver las características del Software Instalado	Evidente
1.5	Permitir ingresar la dirección IP del Computador al cual se observará las características	Evidente
1.6	Permitir observar los procesos en ejecución del Sistema Operativo de un determinado Computador	Evidente

1.7	Permitir Testear un equipo de la Red	Evidente
1.8	Comunicación entre Procesos de verificación	Oculta

5. *ATRIBUTOS*

ATRIBUTO	*DETALLE_ATRIBUTO*
Facilidad de Uso	(Detalle) Fácil de usar por teclado o mouse
Plataforma del Sistema	(Detalle) Sistema Operativo Windows
Tolerancia a fallos	(Restricción de Frontera) No causar daños en caso de falla de Hardware o software
Metáfora de Interfaz	(Detalle) Interfaz amigable para el usuario
Tiempo de Respuesta	(Restricción de Frontera)Respuesta máxima en 5 segundos

6. *ATRIBUTOS DEL SISTEMA EN LAS ESPECIFICACIONES DE FUNCIONES*

Ref.	FUNCIÓN	*CATEGORÍA*	*ATRIBUTO*	*DETALLE ATRIBUTO*	*TIPO*
1.1	Permitir Visualizar datos del Hardware interno de un Computador	Evidente	Fácil de Uso	(Detalle) Fácil de usar por teclado o mouse	Obligatorio
			Metáfora de Interfaz	(Detalle) Interfaz amigable para el usuario	Obligatorio

			Tiempo de Respuesta	(Restricciones de frontera) Respuesta máxima en 5 segundos	Obligatorio
1.2	Permitir Seleccionar un dispositivo especifico de hardware para observar sus características	Evidente	Fácil de Uso	(Detalle) Fácil de usar por teclado o mouse	Obligatorio
			Metáfora de Interfaz	(Detalle) Interfaz amigable para el usuario	Obligatorio
			Tiempo de Respuesta	(Restricciones de frontera) Respuesta máxima en 5 segundos	Obligatorio
1.3	Permitir visualizar el Software instalado actualmente en un Computador	Evidente	Fácil de Uso	(Detalle) Fácil de usar por teclado o mouse	Obligatorio
			Metáfora de Interfaz	(Detalle) Interfaz amigable para el usuario	Obligatorio
			Tiempo de Respuesta	(Restricciones de frontera) Respuesta máxima en 5 segundos	
1.4	Permitir ver las características del Software Instalado	Evidente	Fácil de Uso	(Detalle) Fácil de usar por teclado o mouse	Obligatorio
			Tiempo de Respuesta	(Restricciones de frontera) Respuesta máxima en 5 segundos	
			Metáfora de Interfaz	(Detalle) Interfaz amigable para el usuario	Obligatorio

1.5	Permitir ingresar el nombre del Computador al cual se observará las características	Evidente	Tiempo de Respuesta	(Restricción de Frontera) Respuesta máxima en 5 seg.	Obligatorio
			Metáfora de Interfaz	(Detalle) Interfaz amigable para el usuario	Obligatorio
			Tiempo de Respuesta	(Restricciones de frontera) Respuesta máxima en 5 segundos	Obligatorio
1.6	Permitir observar los procesos en ejecución del Sistema Operativo de un determinado Computador	Evidente	Tiempo de Respuesta	(Restricción de Frontera) Respuesta máxima en 5 segundos.	Obligatorio
			Metáfora de Interfaz	(Detalle) Interfaz amigable para el usuario	Obligatorio
			Tiempo de Respuesta	(Restricciones de frontera) Respuesta máxima en 5 segundos	Obligatorio
1.7	Permitir Testear un equipo de la Red	Evidente	Fácil de Uso	(Detalle) Fácil de usar por teclado o mouse	Obligatorio
			Metáfora de Interfaz	(Detalle) Interfaz amigable para el usuario	Obligatorio
			Tiempo de Respuesta	(Restricciones de frontera) Respuesta máxima en 5 segundos	Obligatorio
			Tiempo de Respuesta	(Restricciones de frontera) Respuesta máxima en 5 segundos	Obligatorio

1.8	Comunicación entre Procesos de verificación	Oculta	Tolerancia a fallos	(Restricción de Frontera) No causar daños en caso de falla de Hardware o software	Obligatorio
			Tiempo de Respuesta	(Restricciones de frontera) Respuesta máxima en 5 segundos	Obligatorio

7. ACTORES

1. Administrador de sistemas

8. PROCESOS

ADMINISTRADOR DE SISTEMAS:

Ejecutar Programa de Monitoreo del Computador

Testear un Computador de la red

9. CASOS DE USO DE ALTO NIVEL

1. ***Caso de Uso:*** Ejecutar programa de monitoreo del Computador

   Actores: Administrador de sistemas (Inicia)

   Tipo: Real, primario

   Descripción: El administrador ejecuta el sistema, para verificar los recursos, tanto hardware como software del Computador

2. ***Caso de Uso:*** Testear un Computador de la red

Actores: Administrador de sistemas (Inicia)

Tipo: Real, Primario

Descripción: El Administrador testea un Computador determinado de la red.

UNIVERSIDAD TÉCNICA PARTICULAR DE LOJA BIBLIOTECA GENERAL

10. CASOS DE USO EXPANDIDOS

1. ***Caso de Uso:*** Ejecutar programa de monitoreo del Computador

Actores: Administrador de Sistemas (Inicia)

Tipo: Real, primario

Resumen: El administrador de sistemas ejecuta el sistema, para verificar los recursos, tanto hardware como software del Computador

Referencias Cruzadas: - Testear un Computador de la red

CURSO NORMAL DE EVENTOS

ACCIÓN DEL ACTOR	RESPUESTA DEL SISTEMA
1. El administrador de sistemas ejecuta el sistema	
	2. El sistema realiza el test al Computador local.

	Computador local.
3. El administrador de sistemas selecciona el dispositivo a analizar del Computador	
	4. El sistema presenta la información del dispositivo seleccionado.
5. El administrador de sistemas visualiza la información del computador.	

2. ***Caso de Uso:*** Testear un equipo de la red

Actores: Administrador de sistemas (Inicia)

Tipo: Real, primario

Resumen: El Administrador de sistemas testea un Computador determinado de la red.

Referencias Cruzadas: - Ejecutar programa de monitoreo del Computador.

CURSO NORMAL DE EVENTOS

ACCIÓN DEL ACTOR	RESPUESTA DEL SISTEMA
1. El administrador de sistemas ingresa en el menú Archivo a la opción Equipo	

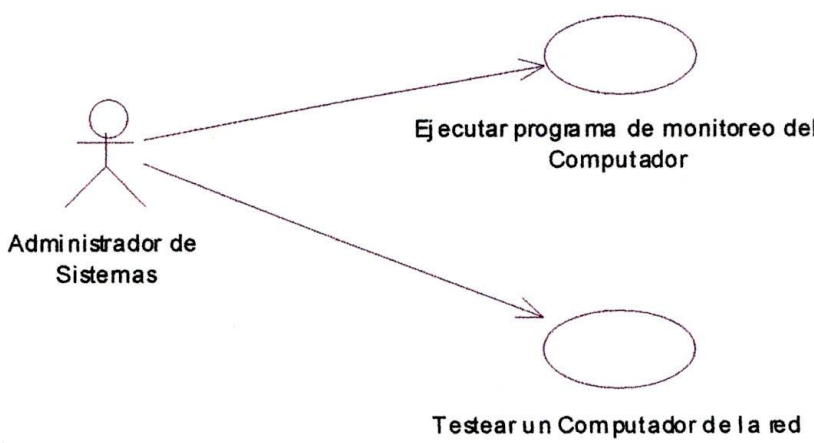
remoto	
	2. El sistema muestra la pantalla para el ingreso del IP.
3. El administrador de sistemas ingresa la dirección IP del Computador que desea testear.	
	4 El sistema busca en la red el Computador solicitado y muestra las características del Computador.
5. El administrador de sistemas observa la información del computador solicitado.	

CURSO ALTERNO

Línea 4: El Computador solicitado no es encontrado. Ingresar otra dirección IP o revisar la conexión de red.

11. DIAGRAMAS DE CASO DE USO

PROCESOS	***ACTORES***	***TIPO DE ACTOR***
Ejecutar programa de monitoreo del Computador Proceso Interno	Administrador de sistemas	Actor Interno
Testear un equipo de la red Proceso Interno	Administrador de sistemas	Actor Interno



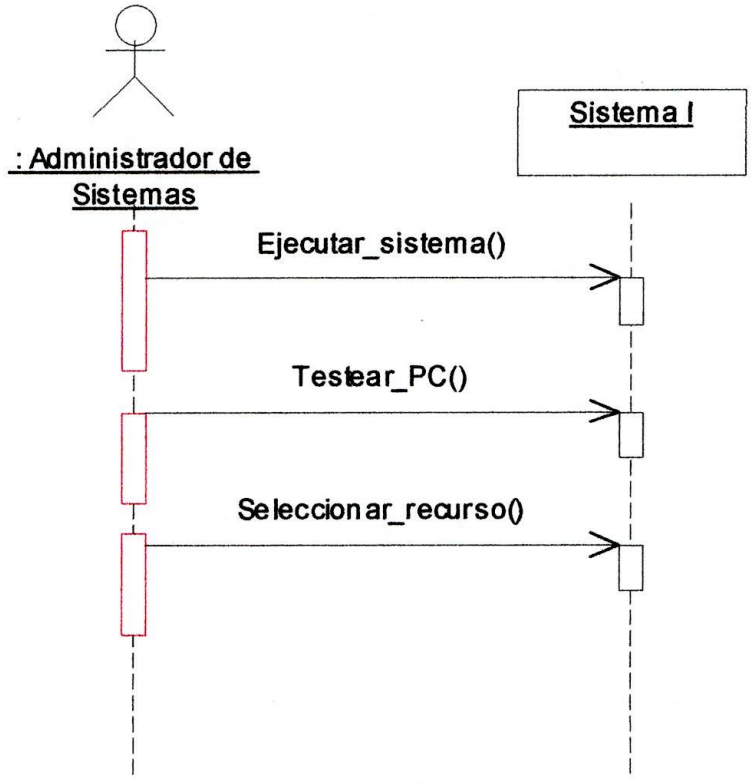
12. CUADRO DE RESUMEN

Nombre del Caso de uso	***Identificación del C. U.***	***Objeto***	***Atributo***
Ejecutar programa de monitoreo del Computador	UCE001	Software	Nombre
			Serie
		Hardware	Nombre
			Serie
Testear un computador de la red	UCE002	Software	Nombre
			Serie
		Hardware	Nombre
			Serie

13. DIAGRAMAS DE SECUENCIA

UCE001 : *Ejecutar programa de monitoreo del Computador*

Actores : Administrador de Sistemas



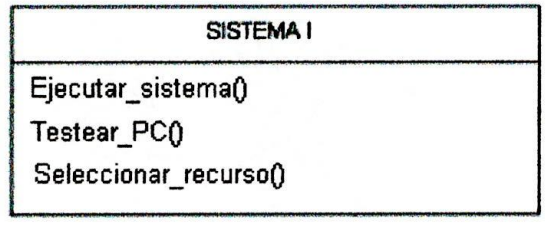
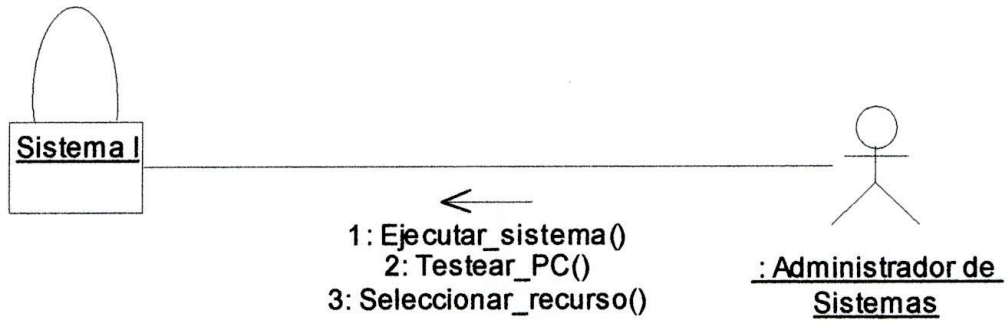
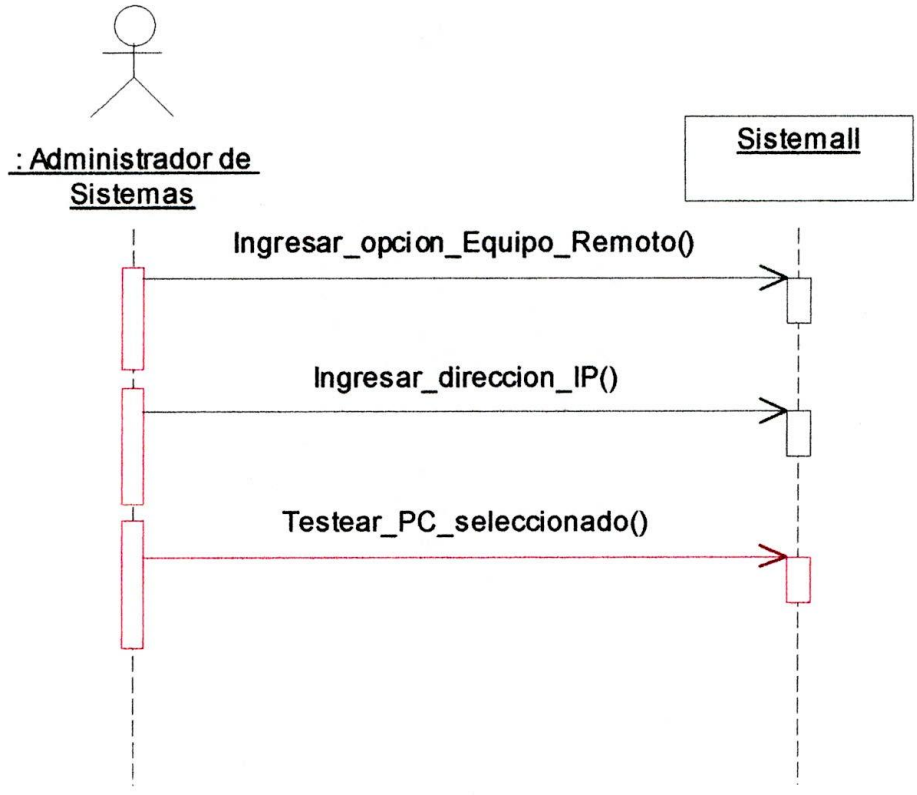


Diagrama de Colaboracion



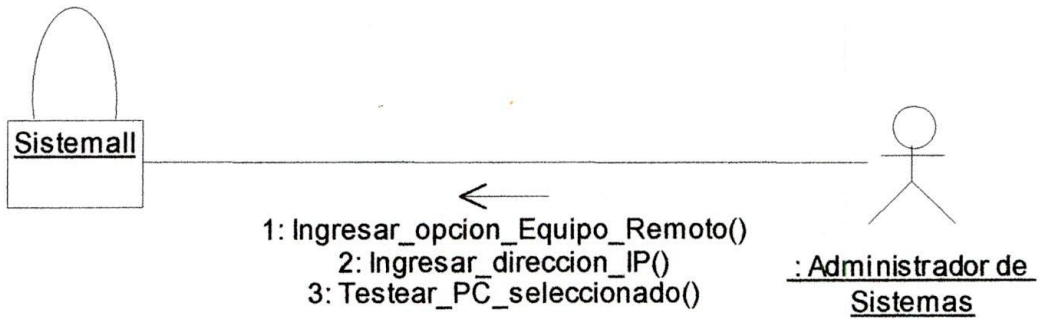
UCE002 : Testear un computador de la red

Actores : Administrador de sistemas



SISTEMA II
Ingresar_opcion_equipo_remoto() Ingresar_direccion_IP() Testear_PC_seleccionado()

DIAGRAMA DE COLABORACION



12. CONTRATOS

NOMBRE: Ejecutar Sistema

RESPONSABILIDAD: Ingresar al sistema

TIPO: Sistema

REFERENCIAS CRUZADAS: Testear un computador de la red

PRECONDICIONES: Tener el sistema instalado en cada Computador de la Red

POSCONDICIONES: Se analiza los recursos del Computador

NOMBRE: Testear Computador

RESPONSABILIDAD:	Analizar los recursos del Computador
TIPO:	Sistema
REFERENCIAS CRUZADAS:	Testear un computador de la red
SALIDA:	Valores de las características software y hardware de Computador
PRECONDICIONES:	Haber ejecutado el sistema.
POSCONDICIONES:	Se muestra listado de componentes software y hardware del Computador.

NOMBRE:	Seleccionar recurso
RESPONSABILIDAD:	Analizar los recursos del Computador.
TIPO:	Sistema
REFERENCIAS CRUZADAS:	Testear un computador de la red
SALIDA:	Valores de recurso seleccionado.
PRECONDICIONES:	Haber ejecutado el sistema.
POSCONDICIONES:	Se muestra valores del recurso seleccionado

NOMBRE:	Ingresar opción equipo remoto
RESPONSABILIDAD:	Ingresar a la opción para acceder a un Computador de la red.
TIPO:	Sistema
REFERENCIAS CRUZADAS:	Ejecutar programa de monitoreo del Computador.
PRECONDICIONES:	Haber ingresado al sistema
POSCONDICIONES:	Se muestrar pantalla para el ingreso de la dirección IP.

NOMBRE:	Ingresar dirección IP
RESPONSABILIDAD:	Ingresar la dirección IP del Computador que se desea testear
TIPO:	Sistema
REFERENCIAS CRUZADAS:	Ejecutar programa de monitoreo del Computador.
NOTAS:	La dirección IP debe tener el siguiente formato: xxx.xxx.xxx.xxx
EXCEPCIONES:	En caso de que La dirección IP sea incorrecta presentar mensaje de "No se puede establecer comunicación con el Computador solicitado"
PRECONDICIONES:	Haber ingresado a la opción equipo remoto El campo dirección IP está enfocado.
POSCONDICIONES:	Se ingresa al Computador seleccionado.

NOMBRE:	Testear Computador seleccionado.
RESPONSABILIDAD:	Testear los recursos del Computador seleccionado.
TIPO:	Sistema
REFERENCIAS CRUZADAS:	Ejecutar programa de monitoreo del Computador.
PRECONDICIONES:	Haber ingresado la dirección IP del Computador requerido.
POSCONDICIONES:	Se presenta los valores de los recursos del Computador seleccionado.

CONCLUSIONES

Al término del presente documento se ha llagado a concluir lo siguiente:

- No podemos utilizar cualquier herramienta (Lenguaje de programación) para poder obtener de una manera correcta la información del hardaware de un PC, ya que se tiene que trabajar en unos casos a bajo nivel (emsamblador), para acceder a direcciones especificas de los registros ya sea del procesador o de la memoria.

- Las caracteristicas del Hardware varian de un fabricante a otro, ya que cada uno tiene una manera diferente de diferenciar su dispositivo, por lo que a veces surgen los conflictos de direcciones de memoria.

- La información del Sistema Operativo es diferente entre cada versión por lo que se debe remitirse a diferentes directorios al momento de querer extraer las características de uno de ellos, lo que dificulta el trabajo de las aplicaciones que se desen ejecutar.

- No todos los programas o aplicaciones poseen una información detallada sobre sus caracteriticas, ya que esta depende en si del fabricante o desarrollador, por lo que no podemos dar una definición exacta sobre cada uno de ellos.

RECOMENDACIONES

Al momento de querer desarrollar una aplicación se este tipo se recomienda:

- Utilizar una herramienta en la cual se pueda programar rutinas a bajo nivel (C, C++, Pascal, Asembler, Delphi).

- No debemos remitirnos unicamente al registro del Sistema Operativo para obtener las caracteriticas Hardware del equipo, ya que aquí no se tiene toda la información que estos poseen, esta solo es superficial.

- Al utilizar el registro del Sistema Operativo, para obtener información primeramente determinar la versión del mismo.

- Revisar el pensun de estudios, ya que actualmente todos los lenguajes de programación se estan orientando únicamente al desarrollo WEB, y se le ha restado importancia a la programación pura, como es el lenguaje Pascal (Delphi) y C o C++, ya que este último es la base del sinmunero de lenguajes y Sistemas Operativos que existen.

- Al momento de adquirir un equipo asegurarnos de resisar que sus series esten marcadas en los dispositivos, ya que cada uno posee un único serial que lo identifican, el mismo que se lo reconfirmará luego mediante software.

ANEXOS

MANUAL DEL USUARIO

REQUISITOS MINIMOS

- Computador Pentium II o superior
- 64 MB RAM de memoria o superior
- 10 MB de espacio en disco
- Windows 98 o Superior
- Microsoft Access (Solo en el Equipo que tenga el servidor)
- Red con protocolo TCP/IP

El sistema se compone de dos partes, un cliente y un servidor, por lo que deberemos elegir que queremos instalar en nuestro equipo.

Instalación del Servidor

Desde la unidad de CD-ROM abrir la carpeta **Servidor** y ejecutar el archivo Setup.exe el cual nos mostrara un asistente el que nos guiara a lo largo de toda la instalación del sistema.

Después que se ha terminado de instalar el sistema, debemos crear una conexión ODBC con el driver de Microsoft Access y que tenga el nombre ***Sisinfo***, la base de datos debe buscarla en el directorio donde hayamos instalado el sistema.

Desde el menú inicio -> programas -> Sisinfo ejecutar el programa Sisinfo, y se abrirá el programa con el cual administraremos todos los equipos en los que hayamos instalado el cliente.

Instalación del Cliente

Desde la unidad de CD-ROM abrir la carpeta **Cliente** y ejecutar el archivo Setup.exe el cual nos mostrara un asistente el que nos guiara a lo largo de toda la instalación del sistema.

Cuando haya finalizado la instalación del cliente, debemos ejecutar el sistema cliente y este se ubicara en forma minimizada en la bandeja del sistema, y mediante él podrá acceder con el programa servidor.

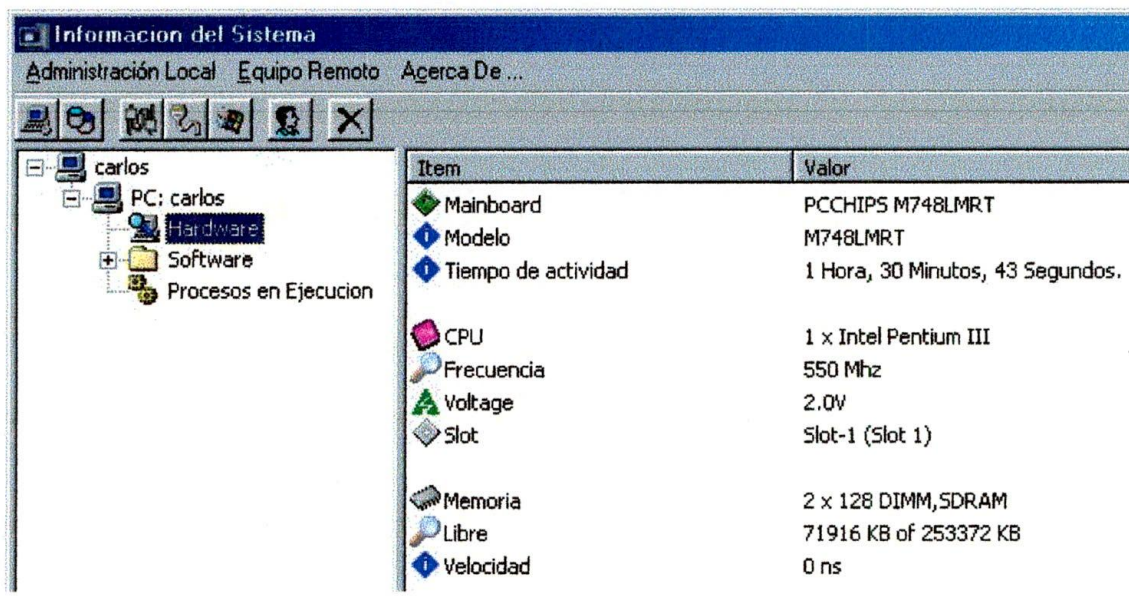
MANEJO DEL SISTEMA SERVIDOR

Información del Sistema

Sistema de Administración de Computadores

Mediante esta herramienta podemos administrar los equipos remotos que posean instalado un programa cliente, el mismo que nos permitirá acceder para poder extraer toda la información del hardware y software que este posea.

La siguiente figura muestra la información local del hardware del computador y a su vez, posee las diferentes opciones de administración de los equipos remotos:



Menú principal: Administración Local, Equipo Remoto

Administración Local

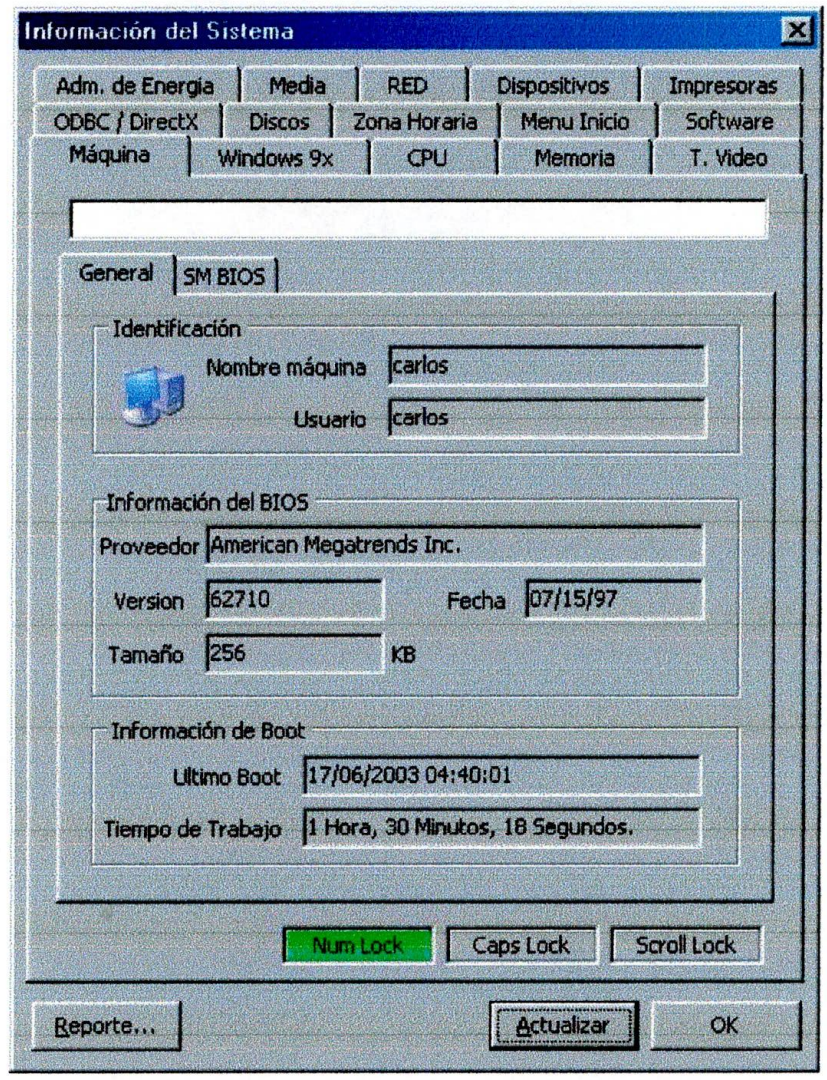
Opción de Administración Local que posee las siguientes opciones:

- ✓ Información Local Completa
- ✓ Administración de Datos

Información Local Completa

Muestra la información local del computador, todos, sus componentes, como son CPU, T. video, T. Sonido, T de Red, etc, todo lo relacionados con el equipo

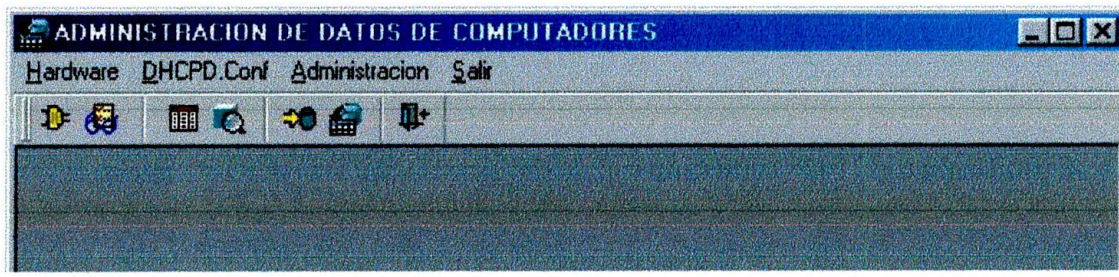
Aquí nos aparece la siguiente ventana:



Administración de Datos

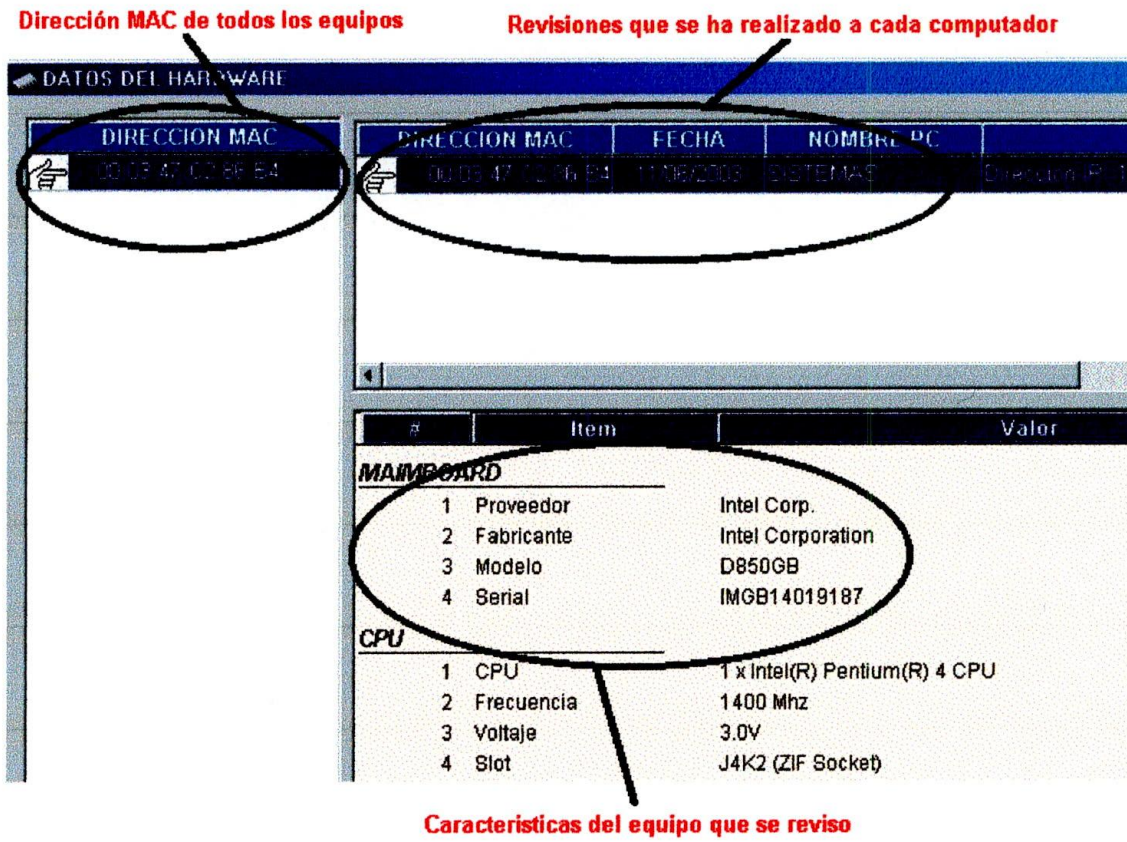
Aquí podemos ver toda la información que se encuentra almacenada en una base de datos, por lo que se nos muestra otra ventana en la que tenemos varias opciones

la pantalla es la siguiente:



Datos Hardware

En esta opción podemos ver toda la información de los diferentes equipos a los que hemos realizado un chequeo y hemos almacenado los datos,



Aquí podemos observar tres secciones:

En la sección de la izquierda se encuentran las direcciones MAC de todas las maquinas que se encuentran dentro de la base de datos

En la sección de la parte superior están toas las revisiones que se ha realizado a los equipos, estar ordenadas por fecha de registro

En la sección de la parte inferior tenemos todos los datos del equipo, de acuerdo a la fecha que escojamos se nos mostrará esta información, aquí podremos observar si algún equipo ha sufrido cambios desde la ultima vez que se lo reviso.

Hardware x Dirección MAC

Aquí podemos realizar la consulta de un determinado equipo ingresando la Dirección MAC del mismo, se nos mostrara, la información de este equipo en las diferentes fechas que se haya realizado la respectiva revisión.

En la parte superior se nos mostrarán las diferentes revisiones que se haya realizado, y en la parte inferior las características que ha tenido en una determinada fecha.

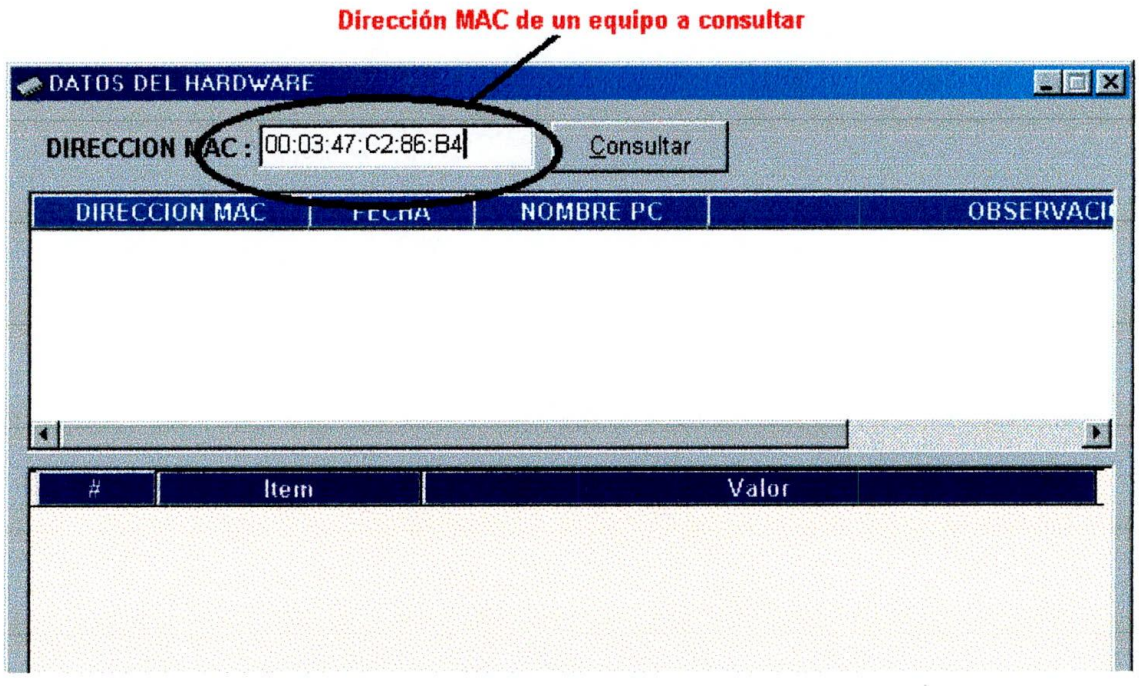


Tabla DHCPD.CONF

Tabla que posee la dirección MAC, dirección IP y nombre del host de todos los equipos que se encuentran dentro de la red.

Esta información se la obtiene del archivo dhcpd.conf, el mismo que es migrado desde un servidor dhcp para ser procesado y poder mostrar estos datos.

Haciendo click en cualesquiera de las columnas ordenaremos los datos por esta.

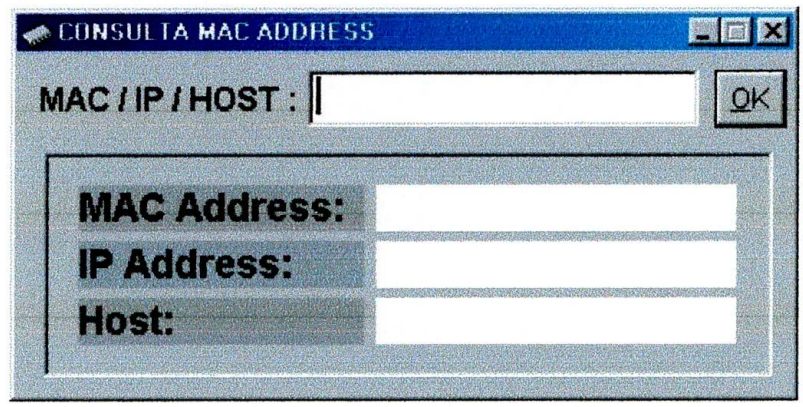
DATOS ARCHIVO dhcpd.conf

#	Address MAC	Address IP	Host
1	00:02:3F:7E:50	172.16.6.31	Vinicio
2	00:02:3F:7F:A8	172.16.6.244	soporte
3	00:02:55:1B:BD:72	172.16.8.124	Goofi-01
4	00:02:55:1D:3A:11	172.16.8.243	Goofi-04
5	00:02:55:1D:57:2E	172.16.8.244	Muti
6	00:02:55:1D:AC:7A	172.16.8.211	JUANCA
7	00:02:55:1D:C6:ED	172.16.3.26	WbMultimedia2
8	00:02:55:1D:CB:1A	172.16.8.246	janneth
9	00:02:55:1D:CF:3F	172.16.3.13	Ultnlnet

Consulta Varios

Mediante esta opción podemos realizar una consulta a la tabla DHCPD.CONF, la consulta la podemos realizar por:

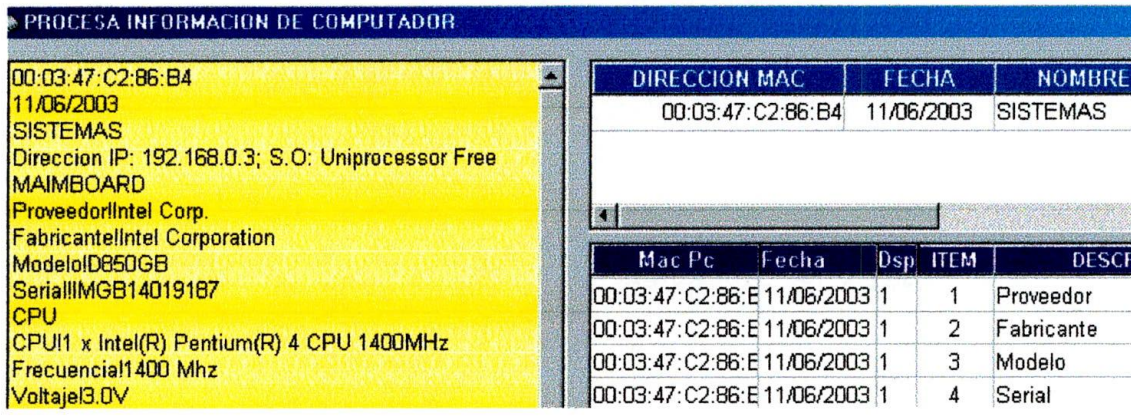
Dirección MAC : XX:XX:XX:XX:XX:XX
Dirección IP : NNN:NNN:NNN:NNN
Host : nombre_pc



Actualiza Datos Equipo

Desde aquí actualizamos los datos de un determinado equipo que estemos realizando la respectiva revisión, estos datos se almacenan en una base de datos la misma que nos servirá para futuras revisiones que necesitemos hacer a los diferentes equipos. Esta actualización la podemos hacer una vez por día.

Aquí únicamente presionamos el botón Procesar para ejecutar un proceso interno que nos hace el trabajo, luego grabamos y entonces se habrán guardado los datos.

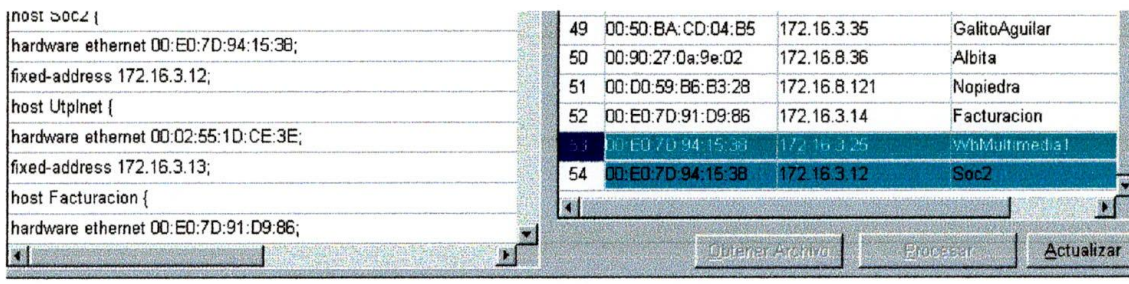


Actualiza Datos DHCP.

Desde aquí nosotros actualizamos los datos de nuestra tabla DHCP.CONF, únicamente debemos hacer click en el botón Obtener Archivo con lo cual nos aparecerá una ventana para conectarnos al servidor dhcp mediante ftp, luego de todo esto hacemos un click en el botón

Procesar para proceder a ejecutar un proceso interno el mismo que nos extraerá toda la información para llenar la tabla.

Si existen direcciones MAC repetidas nos saldrá un mensaje indicándonos sobre el particular, por lo que tendremos que eliminar una de estas presionando el mouse con el botón derecho sobre la fila que seleccionemos.



Finalmente hacemos Click sobre el botón Actualizar para grabar los datos procesados hacia nuestra base de datos, la misma que nos servirá para futuras consultas.

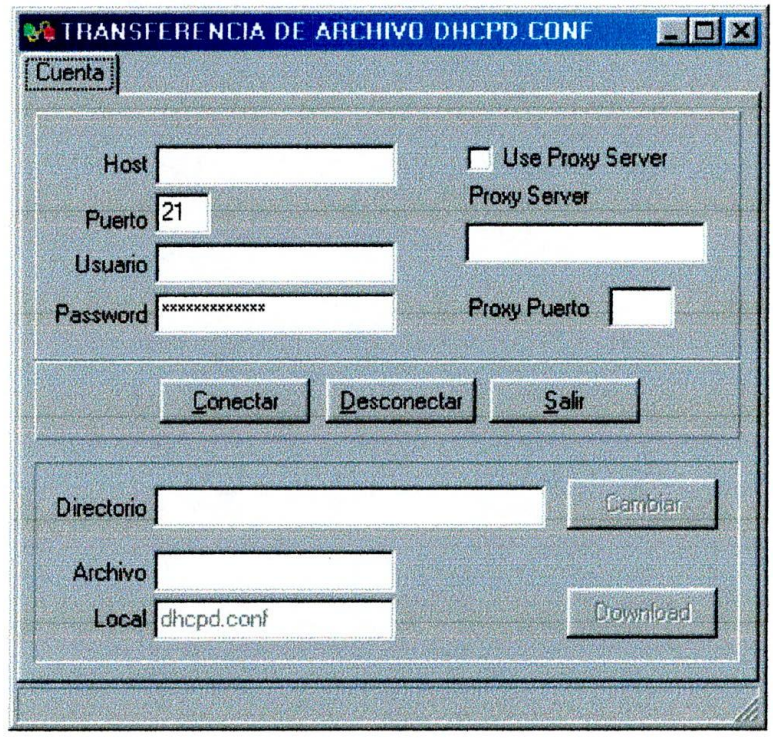
Transferencia de Archivo

En esta pantalla debemos realizar una conexión al servidor dhcp, para poder obtener el archivo dhcp.conf, el mismo que nos servirá para llenar la tabla de direcciones MAC de nuestra base de datos.

Debe llenar los campos de Host, puerto, Usuario que nos hayan asignado y su respectivo password.

En el caso de que el servidor dhcp necesite un servidor proxy debemos marcar la casilla Servidor Proxy y llenar el campo Proxy Server con su respectivo nombre además de su puerto.

Luego de haber llenado estos datos hacemos click en el botón conectar.



En la casilla Directorio debemos poner el PATH donde se encuentra el archivo dhcp.conf, que es el que tenemos que obtener y presionamos el botón Cambiar.
Para después hacer click en download, con lo que completaríamos la descarga del archivo.

Después que se haya hecho esto nos desconectamos y salimos de esta utilidad FTP.

Equipo Remoto

Posee las siguientes sub opciones:

- ✓ Conectar Equipo Remoto
- ✓ Shutdown Equipo Remoto
- ✓ Desconectar

Conectar Equipo Remoto

Mediante esta opción nos podemos conectar a un equipo remoto, el mismo que tendrá instalado el cliente de Sisinfo, el mismo que debe de estar corriendo, aquí nos parece la siguiente ventana como se muestra en la figura

aquí encontramos tres formas de conectarse al equipo remoto:

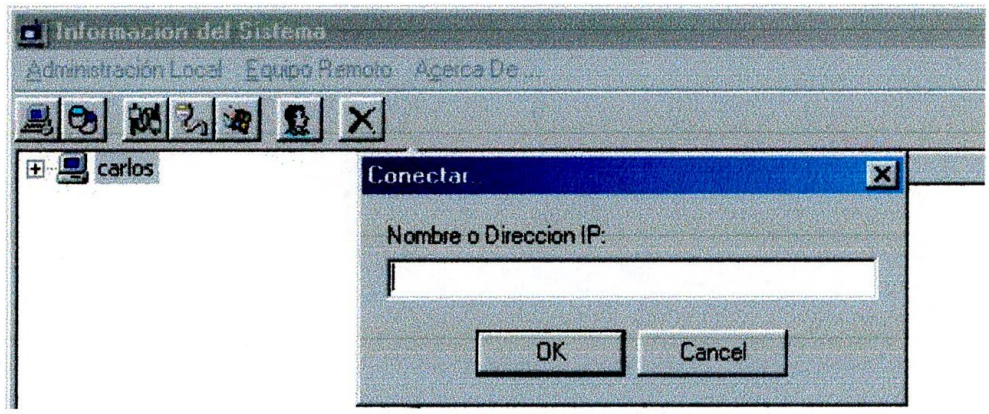
1. Dirección MAC : Dirección física de la tarjeta de Red del equipo tiene el siguiente formato: XX:XX:XX:XX:XX:XX
2. Dirección IP : Dirección asignada el equipo por medio del protocolo TCP/IP formato: XXX.XXX.XXX.XXX
3. Nombre de la MAQUINA : Nombre que posee la maquina (nombre del Computador)

Tenemos que seleccionar con cual de estas opciones nos conectamos, luego poner en el campo inferior el valor correspondiente de acuerdo al ítem seleccionado:

Finalmente presionar OK

Shutdown Equipo Remoto

En esta opción nos aparece la pantalla siguiente:



con la que podemos apagar el equipo remoto, para lo cual debemos de conectarnos al mismo y enviarle la orden

Desconectar Equipo Remoto

Aquí nos desconectamos del equipo remoto que estábamos viendo la información, y solo administraremos localmente toda la información que hemos obtenido.

BIBLIOGRAFÍA

- CRAIG Stinson. Guía Completa de Microsoft Windows 98, Editorial McGraW-Hill, México 1999.

- CRAIG Stinson. RUNING Microsoft Windows 98, Editorial McGraW-Hill, Interamericana de España 1999.

- MUELLER Scott. Manual de Actualización y Reparación de PCs, Editorial Prentice Hail, México 1998.

- CHARTE OJEDA Francisco, Programación en Windows 2000, Editorial Anaya Multimedia, Madrid 2000.

- MICROSOFT, Corporation. MSDN Library Visual Studio 6.0.

- HALSALL Fred. Comunicación de Datos, Redes de Computadores y Sistemas Abiertos. Cuarta edición. Editorial Addison-Wesley Iberoamericana. Wilmington, Delaware, EUA. 1998.

- DEITEL H. M. Sistemas Operativos. Segunda edición. Editorial Addison-Wesley Iberoamericana. Wilmington, Delaware, EUA. 1993.

- ANGULO José M., FUNKE Enrique M. 386 y 486 Microprocesadores Avanzados de 32 Bits. Editorial Paraninfo Madrid, 1992.

- CANTÚ Marco, La Biblia de DELPHI 5.0 Editorial Anaya Multimedia Madrid 2000.

INDICE

TITULO	PAG.
CERTIFICACIÓN	i
CESION DE DERECHO	ii
AGRADECIMIENTO	iii
AUTORIA	iv
DEDICATORIA	v
TITULO	vi
ESQUEMA DE CONTENIDOS	vii
PROLOGO	ix
INTRODUCCIÓN	x
OBJETIVOS	xii
CAPITULO I.	
1. DETERMINACIÓN DE LOS RECURSOS HARDWARE	1
1.1 Verificación de tipo de procesador	1
1.2 Determinación de la cantidad de memoria en el computador	9
1.3 Tamaño y espacio del Disco Duro	14
1.4 Tarjetas del computador	20
1.4.1. Tarjeta de Video	20
1.4.2. Tarjeta de RED	23
1.4.3. Tarjeta de Sonido	26
1.4.4. Fax Moden	28
1.5 Registro de datos observados	30
CAPITULO II.	
2. MONITOREO DE LOS RECURSOS SOFTWARE	34
2.1 Aplicaciones en ejecución	34
2.2 Recursos utilizados por las aplicaciones	40
2.3 Liberación de recursos	43

CAPITULO III.

45

3. VERIFICACIÓN DE SOFTWARE INSTALADO 45

3.1 Registro del Sistema Operativo 46

Descripción detallada de claves 48

Subclaves Principales 53

3.2 Detección de programas instalados 57

3.3 Verificación de características de los programas.

CAPITULO IV

60

4. VERIFICACIÓN DE SOFTWARE INSTALADO 60

Análisis del Sistema

CONCLUSIONES 75

RECOMENDACIONES 76

ANEXOS 77

BIBLIOGRAFIA 78

INDICE 79