ESCUELA POLITÉCNICA DEL EJÉRCITO

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA, REDES Y COMUNICACIÓN DE DATOS

PROYECTO DE GRADO PARA LA OBTENCIÓN DEL TÍTULO DE INGENIERÍA

DISEÑO E IMPLEMENTACIÓN DE UNA PLATAFORMA E-LEARNING PARA LA MATERIA DE TECNOLOGÍAS DE SOFTWARE PARA ELECTRÓNICA.

BYRON ORLANDO DELPINO GUADALUPE

Sangolquí – Ecuador

2013

CERTIFICACIÓN

Certificamos que el presente proyecto de grado titulado: “DISEÑO E IMPLEMENTACIÓN DE UNA PLATAFORMA E-LEARNING PARA LA MATERIA DE TECNOLOGÍAS DE SOFTWARE PARA ELECTRÓNICA”, bajo nuestra dirección, ha sido desarrollado en su totalidad por el Sr. Byron Orlando Delpino Guadalupe con C.I: 171923003-7.

Atentamente

Ing. Darwin O. Alulema F., MSc.
DIRECTOR

Ing. José Sancho A., Msc
CODIRECTOR

RESUMEN

El presente proyecto de grado abarca el proceso de diseño e implementación de una plataforma e-learning para la materia de tecnologías de software para electrónica empleando el lenguaje de programación java.

Inicialmente se desarrolla los contenidos sobre nuevas tecnologías java como el manejo del puerto usb, programación JavaServer Faces y Java Micro Edition.

Cada tema revisado en la plataforma virtual está conformado por fundamento teórico, videos explicativos, sitios de descargas, laboratorios resueltos en Netbeans y Eclipse, cuestionarios y productos de la unidad, de esta forma se busca que el estudiante integre las nuevas tecnologías para la resolución de problemas relacionados a la ingeniería electrónica.

El aula virtual es implementada en un sitio web que tenga instalado el Sistema de Gestión de Aprendizaje Moodle, donde se cargarán los contenidos desarrollados.

Posteriormente se realizan las pruebas como docente y estudiante con la finalidad de garantizar el correcto acceso y funcionamiento de la plataforma.

Finalmente se propone una guía metodológica para la implementación de una plataforma e-learning, así como una propuesta de plan microcurricular para la materia de Tecnologías de Software para Electrónica, que incluya el Puerto USB, JavaServer Faces y Java Micro Edition.

DEDICATORIA

A creador del Universo, por su infinito amor, su protección, y su presencia.

A mis padres Orlando y Susana, quienes con su apoyo me inculcaron el amor a Dios, la disciplina y el trabajo.

A mi hermana Johanna por su cariño y confianza en todo momento.

A mis familiares, amigos y maestros por acompañarme en la consecución de este objetivo.

A todas las personas quienes han formado parte importante en esta etapa de mi vida.

AGRADECIMIENTO

A Dios, por ser lo más importante en mi vida, por darme la vida, la salud. Por estar siempre con aquel niño, con aquel joven que ha tenido errores y caídas, pero siempre se ha levantado por Tu amor y confianza.

A mis padres por su amor, sus consejos y apoyo incondicional. Los amo.

A mi hermana por ser un aliento, por estar en aquellos momentos cuando la tristeza, el fracaso se querían imponer en mi vida, por esas peleas y discusiones que siempre terminaban en abrazos.

A mis maestros, por sus valiosos conocimientos y experiencias que me han permitido crecer como ser humano y profesional.

A mis amigos, por su lealtad, su amistad perdurará por siempre.

A los Ingenieros Darwin Alulema y José Sancho por su asesoramiento y paciencia en el desarrollo de este Proyecto de Grado.

PRÓLOGO

El crecimiento de la informática y las telecomunicaciones han permitido la creación de herramientas que mejoren el proceso de enseñanza-aprendizaje en la formación profesional. En la actualidad el aprendizaje electrónico (e-learning) es una excelente alternativa para la adquisición de conocimientos como apoyo tecnológico, siempre seleccionando información calificada, pertinente y bajo estándares de certificación del conocimiento.

E-learning se presenta como un instrumento relevante para la enseñanza de cualquier temática, más aun en cuanto al impartir conocimientos sobre programación JAVA, siendo un soporte y complemento para docentes y alumnos que la utilicen como un elemento de consulta e investigación.

El presente proyecto de grado consiste en el desarrollo de una plataforma e-learning que abarque temas relacionados con la carrera de Ingeniería Electrónica, los contenidos de esta aula virtual engloban la comunicación a través del puerto usb, programación de aplicaciones cliente-servidor mediante JavaServer Faces y el desarrollo de aplicaciones en dispositivos móviles con Java Micro Edition.

En la plataforma e-learning, el estudiante contará con documentos, videos explicativos, laboratorios resueltos, así como cuestionarios y actividades a desarrollar, que facilitarán un aprendizaje acorde a la enseñanza universitaria de primer nivel, como una referencia básica que es necesario proyectarla para posteriores estudios o proyectos que puedan contemplarse.

Gracias a la utilización de una plataforma E-learning para la materia de Tecnologías de software para electrónica, el alumno desplegará múltiples capacidades en cuanto al uso de JAVA como un software libre con grandes ventajas, elevando sus posibilidades de resolver problemas concernientes a su carrera.

INDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO 1

FUNDAMENTO TEÓRICO

PUERTO USB

1.1. Historia del puerto USB

En un principio los puertos serial, paralelo y PS/2 eran los únicos medios que permitían conectar a la PC con impresoras, mouse y teclados. Las velocidades en que se transmitían y recibían datos entre las dos partes no superaban el 1Mbps.

En 1995 ante la necesidad de crear una interfaz estandarizada que ofrezca comunicaciones a gran velocidad entre la computadora y los distintos periféricos, las empresas Intel, IBM, Northern Telecom, Compaq, Microsoft, Digital Equipment Corporation y NEC conforman USB-IF (USB Implementers Forum) para crear el puerto USB.

- **USB 1.0**

En enero de 1996 USB-IF implementa USB 1.0, la cual presenta dos tipos velocidades, la primera Low Speed con 1.5 Mbps y la segunda denominada Full Speed con 12Mbps de velocidad.

En 1998, Apple lanza al mercado el primer computador con puerto USB, el iMac, el cual permitía conectar el teclado y mouse. La primera versión de Windows que soportaba USB fue Windows 95 OSR 2.1.

- **USB 2.0**

En abril del 2000, Hewlett Packard, Philips y Lucent se unen a la corporación USB-IF, y desarrollan la versión USB 2.0 la cual presenta velocidades de hasta 480 Mbps. USB 2.0 es compatible con USB 1.0 en velocidad y conectores. USB 2.0 es la versión más difundida en la actualidad.
En cuanto a servidores, el sistema operativo Windows 2000 ya soporta USB 2.0.

- **USB 3.0**

En diciembre del 2008, USB-IF crean USB 3.0, la cual presenta mejoras en velocidad de transmisión de hasta 5 Gbps, reduce el consumo de energía y es compatible con las versiones anteriores de USB. En esta versión se incrementa la corriente de 500 a 900mA.

El primer sistema operativo que soportó USB 3.0 fue Linux Kernel 2.6.31.

USB es una tecnología que corrige las limitaciones del puerto serial y paralelo ya que presenta mayores velocidades de transmisión, prestaciones en cuanto a consumo de energía y dispositivos que se pueden interconectar, gracias al desarrollo tecnológico en la última década USB está presente en equipos informáticos, electrodomésticos, vehículos.

Interfaz	Mbps
Paralelo	0,3
Serial	0,46
Ps/2	0,15
USB1.0 Low	1,5
USB 1.0 Full	12
USB 2.0	480
USB 3.0	5000

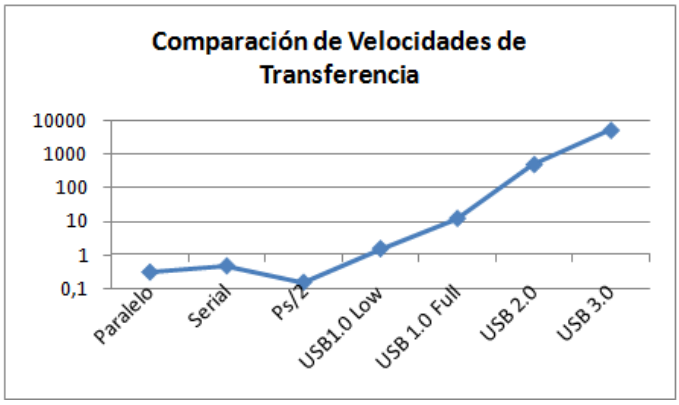


Figura 1. 1. Comparación de Velocidades de Transferencia Interfaces

Autor: Byron Delpino

1.2. Conector USB

Todas las versiones de USB básicamente presentan tres tipos de conectores, el conector tipo A, el B, y tipo Mini la cual está incluida para conectar dispositivos pequeños (cámaras, celulares). Su diferencia radica en la forma del conector y en la posición de cada uno de sus pines, más las características eléctricas y protocolos de transmisión de datos no varían. La versión USB 3.0 presenta mayor cantidad de pines de transmisión y recepción los cuales permiten una mayor tasa de transferencia.

Los tipos de conectores USB presentan dos variaciones, macho o plug y hembra o recipiente.

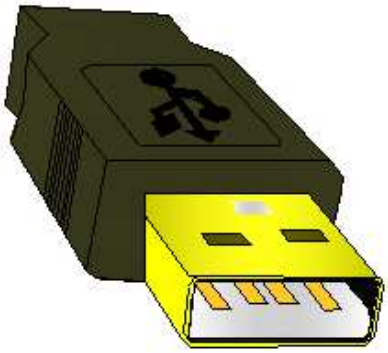


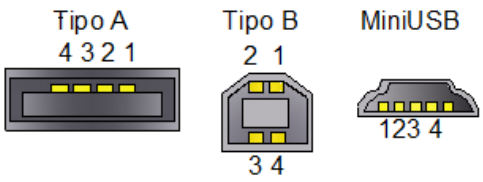
Figura 1. 2. Estructura Física de un Conector USB

Autor: Byron Delpino

El conector tipo A es de forma rectangular (16x11.75 mm), conformado por los pines que dan la alimentación, transmiten y reciben datos, este conector está alojado en el computador.

El conector tipo B es cuadrado (11.5x11.75 mm), el cual presenta los mismo pines de conexión salvo que se utiliza para conectar al computador con algún dispositivo externo (impresoras, módems, scanner).

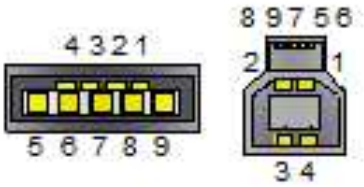
El conector tipo C es de tipo trapezoidal, es útil para conectar al ordenador con cámaras, reproductores de MP3, celulares.



Pin	
1	Alimentación 5V
2	Datos -
3	Datos +
4	Tierra

Figura 1. 3. Pines conector usb 2.0

Autor: Byron Delpino



Tipo A Tipo B

Pin	
1	Alimentación 5V
2	Datos -
3	Datos +
4	Tierra
5	Rx -
6	Rx +
7	Tierra- Drenado
8	Tx -
9	Tx +

Figura 1. 4. Pines conector usb 3.0

Autor: Byron Delpino

1.3. Características de Transmisión

El puerto USB suministra un voltaje nominal de 5V, lo que permite alimentar dispositivos con un bajo consumo de potencia. USB-IF exige como voltaje de alimentación entre 4.375-5.25v los cuales deben ser regulados en 5V.

La corriente es de 500 mA para las 2 primeras versiones USB y un incremento a 900 mA para USB 3.0, esto facilita conectar una mayor cantidad de dispositivos al ordenador. Los dispositivos que superan el consumo permitido por USB requieren su propia fuente de poder.

La distancia máxima en que se puede enviar datos desde un host a un dispositivo va de los 3 a 5m.

La transmisión de datos es binaria ceros y unos, el voltaje para transmitir un bajo (cero) varía entre 0-300mV y un alto (uno) va desde 2.8-3.6V (USB 1.0) y 400mV (USB2.0).

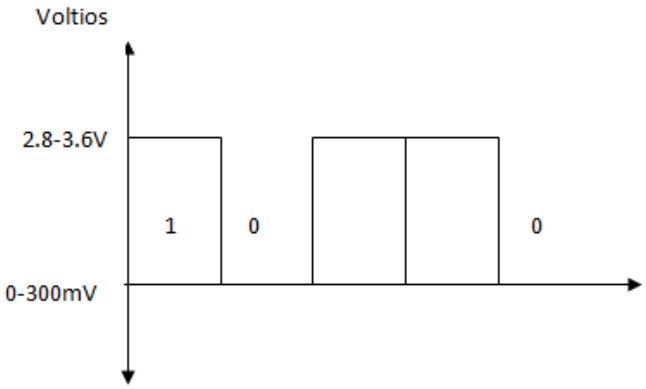


Figura 1. 5. Transmisión de bits puerto USB

Autor: Byron Delpino

La comunicación entre el host y un dispositivo final se da a través de canales lógicos que además de permitir la transmisión y recepción de bytes permite al host asignar un número identificador al dispositivo que servirá para verificar su estado y configuración.

Los canales lógicos se clasifican en cuatro categorías:

- **Transferencias de control -** Envío de comandos.
- **Transferencias isócronas -** Proveen un ancho de banda definido. Usado típicamente para transmisiones en tiempo real (voz, video).
- **Transferencias Interruptoras** – Utilizado para dispositivos que requieren una respuesta rápida por ejemplo, mouse, teclados.
- **Transferencias Masivas -** Para transferencias grandes (archivos) donde se usa todo el ancho de banda del canal.

1.4. API jPicUSB

La API JPicUSB es una clase desarrollada en enero del 2009 que a través de métodos nativos permite conectar a la PC con microcontroladores a través del puerto USB. Esta clase contiene métodos nativos que invocan a determinadas funciones dentro de una librería de vínculo dinámico (.dll) donde se encuentran implementados los métodos de la API de Microchip jUSB.

Esta clase está conformada por:

- ✓ jpicusb.jar: Interfaz java que invoca a la librería dinámica jpicusb.dll
- ✓ jpicusb.dll: Almacena todas las funciones de la API de Microchip (mpusbapi).

1.4.1. Métodos de jpicusb.jar

La interfaz jpicusb.jar posee la clase iface en donde se implementa tres tipos de métodos:

- ✓ ***Métodos de inicialización***.- Estos permiten invocar a la librería dinámica jpicusb.dll y establecer parámetros de configuración como número de dispositivo, canal lógico.
- ✓ ***Métodos generales***.- Permiten abrir o cerrar la conexión con un dispositivo USB, su lectura o escritura incluyendo parámetros como VIDyPID(identificador del dispositivo USB), utilización del canal lógico.
- ✓ ***Métodos rápidos***.- Son métodos semejantes en cuanto al funcionamiento de los Métodos de lectura y escritura, su diferencia radica en que se omite parámetros de configuración del dispositivo (VIDyPID, canales lógicos).

- **Métodos de Inicialización**

  - **Load :** Llama a la librería dinámica jpicusb.dll.

    public static void load()

  - **set_vidpid** : Configura un identificador VIDyPID por defecto al dispositivo USB.
    set_vidpid(java.lang.String s)
    donde:
    s : Un string que contiene el identificador VIPyPID del dispositivo. El formato es "vid_xxxx&pid_yyyy".

  - **set_instance** : Configura una instancia (número de solicitud) por defecto para el dispositivo.

    public static void set_instance(int i)

donde:

i : Número de instancia del dispositivo.

Una vez que se haya configurado parámetros como el VIDyPID y la instancia del dispositivo se puede utilizar los métodos rápidos.

- **Métodos Generales**

  - **GetDeviceCount**: Obtiene el número de dispositivos que coinciden con el identificador VIDyPID.

    public static int GetDeviceCount(java.lang.String pVIDyPID)

    donde:

    pVIDyPID: String que contiene el VIDyPID del dispositivo. El formato es "vid_xxxx&pid_yyyy".

  - **Open**: Abre un canal lógico (pipe) con aquel dispositivo que coinciden su identificador VIDyPID.

    public static long Open(int instance, java.lang.String pVIDyPID, java.lang.String pEP, int dwDir, int dwReserved)

    donde:

    instance : Número de instancia del dispositivo.

    pVIDyPID: String que contiene el VIDyPID del dispositivo. El formato es "vid_xxxx&pid_yyyy".

    pEP: Un string que contiene el número del dispositivo. El formato es "\\MCHP_EPz", z es el número del dispositivo en notación decimal.

    dwDir: Entero que especifica la dirección del dispositivo, sea lectura o escritura.

    dwReserved: Parámetro reservado para uso futuros.

    El método retorna un valor tipo long que representa al controlador o manejador (handle) del canal lógico del dispositivo.

  - **Close**: Cierra el canal lógico del dispositivo que posea un determinado handle o controlador.

    public static boolean Close(long handle)

donde:

handle: Controlador o manejador del canal lógico del dispositivo.

El método retorna un valor tipo booleano que indica si se cerró o no la conexión del canal lógico.

- **Read**: Lee datos del canal lógico de un dispositivo mediante sus controlador (handle).

  public static byte[] Read(long handle, int dwLen, long dwMilliseconds)

  donde:

  handle: Controlador o manejador del canal lógico del dispositivo.

  dwlen: Cantidad de bytes a leer.

  dwMilliseconds: Tiempo de espera en milisegundos.

  El método retorna un vector tipo byte donde se almacena los datos recibidos.

- **Read**: Lee datos del canal lógico de un dispositivo con un determinado identificador VIDyPID.

  public static byte[] Read(java.lang.String pVIDyPID, int instance, int dwLen, long dwMilliseconds)

  donde:

  pVIDyPID: String que contiene el VIDyPID del dispositivo. El formato es "vid_xxxx&pid_yyyy".

  instance : Número de instancia del dispositivo.

  dwlen: Cantidad de bytes a leer.

  dwMilliseconds: Tiempo de espera en milisegundos.

  El método retorna un vector tipo byte donde se almacena los datos recibidos.

- **Write**: Envía datos a través del canal lógico a un dispositivo mediante sus controlador (handle).

  public static long Write(long handle, byte[] pData, int dwLen, long dwMilliseconds)

  donde:

  handle: Controlador o manejador del canal lógico del dispositivo.

  pData: Arreglo de bytes a ser enviados.

dwlen: Cantidad de bytes a enviar.

dwMilliseconds: Tiempo de espera en milisegundos.

El método retorna un valor tipo long que señala el número de bytes escritos.

- **Write**: Envía datos a través del canal lógico de un dispositivo con un determinado identificador VIDyPID.

  public static long Write(java.lang.String pVID_PID, int instance, byte[] pData, int dwLen, long dwMilliseconds)

  donde:

  pVIDyPID: String que contiene el VIDyPID del dispositivo. El formato es "vid_xxxx&pid_yyyy".

  instance : Número de instancia del dispositivo.

  pData: Arreglo de bytes a ser enviados.

  dwlen: Cantidad de bytes a enviar.

  dwMilliseconds: Tiempo de espera en milisegundos.

  El método retorna un valor tipo long que señala el número de bytes escritos.

- **WriteRead**: Envía y recibe datos abriendo dos canales lógicos para un dispositivo con un determinado identificador VIDyPID.

  public static byte[] WriteRead(java.lang.String pVID_PID, int instance, byte[] pData, int dwLenWrite, int dwLenRead, long dwMilliseconds)

  donde:

  pVIDyPID: String que contiene el VIDyPID del dispositivo. El formato es "vid_xxxx&pid_yyyy".

  instance : Número de instancia del dispositivo.

  pData: Arreglo de bytes a ser enviados.

  dwLenWrite: Cantidad de bytes a ser enviados.

  dwLenRead: Cantidad de bytes a recibir.

  dwMilliseconds: Tiempo de espera en milisegundos.

  El método retorna un vector tipo byte donde se almacena los datos recibidos.

- **Métodos Rápidos**

  - **QRead**: Lee datos del canal lógico de un dispositivo con un VIDyPID establecido por defecto a través del método set_vidpid.
    public static byte[] QRead(int dwLen, long dwMilliseconds)
    donde:
    dwlen: Cantidad de bytes a leer.
    dwMilliseconds: Tiempo de espera en milisegundos.
    El método retorna un vector tipo byte donde se almacena los datos recibidos.

  - **QWrite**: Envía datos a través del canal lógico a un dispositivo con un VIDyPID establecido por defecto
    public static long QWrite(byte[] pData,int dwLen, long dwMilliseconds)
    donde:
    pData: Arreglo de bytes a ser enviados.
    dwlen: Cantidad de bytes a enviar.
    dwMilliseconds: Tiempo de espera en milisegundos.
    El método retorna un valor tipo long que señala el número de bytes escritos.

  - **QWriteRead** : Envía y recibe datos abriendo dos canales lógicos para un dispositivo con un identificador VIDyPID por defecto.
    public static byte[] QWriteRead(byte[] pData, int dwLenWrite, int dwLenRead, long dwMilliseconds)
    donde:
    pData: Arreglo de bytes a ser enviados.
    dwLenWrite: Cantidad de bytes a ser enviados.
    dwLenRead: Cantidad de bytes a recibir.
    dwMilliseconds: Tiempo de espera en milisegundos.
    El método retorna un vector tipo byte donde se almacena los datos recibidos.

1.5. Microcontroladores con puerto USB

Los microcontroladores que soportan comunicación USB con el computador van desde la familia 16, 18 y 24xxx. En la siguiente tabla se detallan las características USB en cuanto a memoria, identificador de fabricante y producto de los pics más conocidos en el mercado.

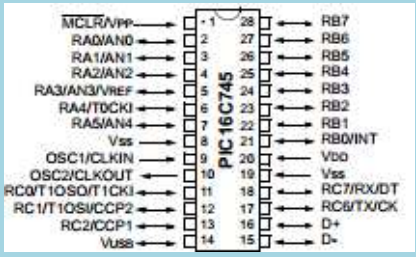
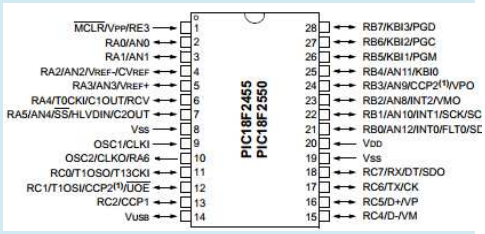
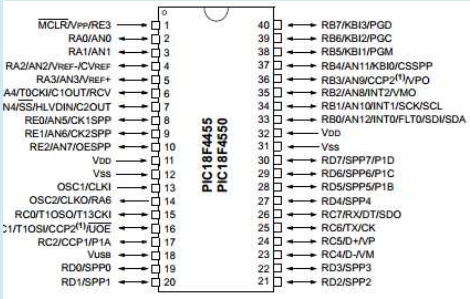
PIC	# Pines	Memoria RAM para USB	Velocidad Mbps para Transferencias Masivas	VID	PID	Diagrama de Pines
16C745	28	64B	1.5	0461	0001	
18F2455	28	1KB	12	04D8	000b	
18F2550	28	1KB	12	04D8	0011	
18F4455	40/44	1KB	12	04D8	000b	
18F4550	40/44	1KB	12	04D8	000b	

Tabla 1. 1. Microcontroladores USB

Autor: Byron Delpino

Fuente: www.microchip.com

1.6. Aplicaciones y Set de instrucciones del Microcontrolador

Para la programación de microcontroladores se cuenta con una variedad importante de compiladores en el mercado. Se distingue dos tipos de lenguajes, los de bajo nivel o aquellos que utilizan un lenguaje ensamblador y los de alto nivel cuyas instrucciones son legibles para el usuario.

El compilador PICC es un lenguaje de alto nivel que permite la programación de microcontroladores desde un nivel cercano al programador, así como la manipulación en aspectos cercanos al hardware del dispositivo.

Este compilador presenta las siguientes ventajas:

- *Fácil programación*.- PICC Compiler dispone de un conjunto de operadores e instrucciones que permiten un tiempo de programación mucho menor que la programación en lenguaje ensamblador.
- *Portabilidad entre sistemas*.- Existe independencia entre el software con respecto al hardware, de esta forma se puede programar en cualquier tipo de plataforma y dispositivo.
- *Desarrollo y ejecución de programas estructurados*.- PICC ya cuenta con una biblioteca de funciones precompiladas listas para ser utilizadas nuevamente.
- *Inserción de código ensamblador*.

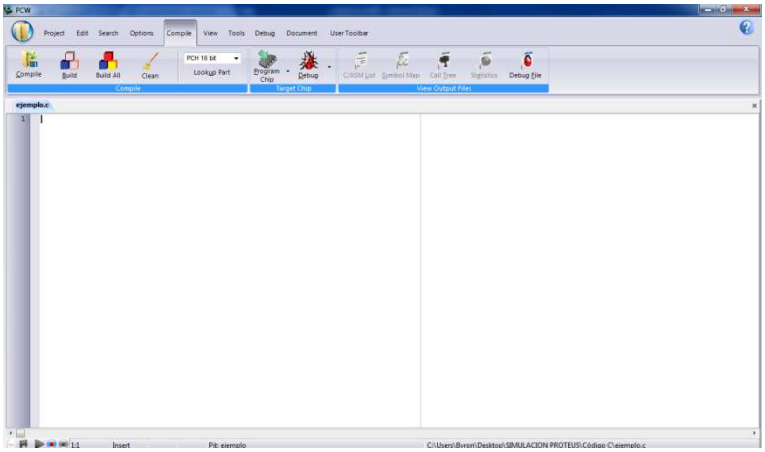


Figura 1. 6. Compilador PICC

Autor: Byron Delpino

Un Microcontrolador al poseer memorias (RAM, ROM), puerto de comunicaciones, convertidores Análogico/Digitales lo hace útil para cualquier tipo de aplicación que involucre el manejo de dispositivos (displays, motores, parlantes), monitoreo de entradas (sensores, teclados), comunicación con microcontroladores o pcs a través de sus puertos serial, USB, Ethernet.

En la actualidad los microcontroladores están implementados en:

- Electrodomésticos: TV, grabadoras, equipos de sonidos, teléfonos, refrigeradoras.
- Equipos de computación: Mouse, teclado, impresoras.
- Industria Automotriz: Mando de sistemas de vehículos (inyección, encendido).
- Industria Militar: Aviones, tanques de guerra, plataformas de disparo, radares.
- Domótica: Sistemas de seguridad, vigilancia, alarmas, ascensores, calefacción.
- Electromedicina: Equipos de radiografía, electrocardiógrafo, tensiómetros.
- Robótica: Brazos robóticos, sensores de movimiento, motores.
- Entretenimiento: Consolas de video, procesamiento de imágenes, manejos de joystick.

El set de instrucciones de un PIC se destaca a continuación:

- **usb-init:** Inicializa el dispositivo USB.

- **usb_task:** Prepara al dispositivo USB.

- **usb_wait_for_enumeration:** Instrucción que ingresa a un bucle hasta que el host haya dado un número de identificación al Microcontrolador.

- **usb_enumerated:** Esta instrucción permite verificar si el dispositivo ha sido numerado por la PC permitiendo el envío y recepción de paquetes.

- **usb_kbhit:** Este método indica si el dispositivo tiene algún dato por leer y ser colocado en el buffer de recepción.

- **usb_put_packet:** Permite el envío de paquetes los cuales se posicionarán en el buffer desde un punto final especificado.
  usb_put_packet(buffer,Datos,Longitud de datos, Toggle)
  donde:
  buffer: Punto final especificado, desde este punto del buffer se envían los paquetes.
  Datos: Los bytes a ser enviados.
  Toggle: Permite la sincronización entre la PC y el Microcontrolador.

- **usb_get_packet:** Recepción de paquete quienes se posicionarán en el buffer desde un punto final especificado.
  usb_get_packet(buffer,Variable,Longitud de datos)
  donde:
  buffer: Punto final especificado, desde este punto del buffer se envían los paquetes.
  Variable: Almacena los datos recibidos

Para programar USB en microcontroladores se requiere incluir una serie de ficheros los cuales incluyen además de las instrucciones previamente analizadas, configuraciones del Microcontrolador, VIDyPID.

- **PICUSB.h:** Cabecera que incorpora las funciones del Microcontrolador USB. Ejm 18F4550.h.

- **pic18_usb.h:** Esta cabecera incluye la configuración de parámetros como velocidad del dispositivo, buffer, sincronización de envío de paquetes.

- **usb.c y usb.h:** Definen las instrucciones para el inicio, enumeración y transmisión de datos del dispositivos usb. Ejm (usb_init, usb_get_packet).

1.7. Laboratorios

1.7.1. Instalación y utilización del puerto USB Virtual de Proteus.

Proteus es un paquete informático creado por Labcenter Electronics que permite el diseño, construcción y simulación de circuitos eléctricos.

Este software consta del Ares, utilizado para la fabricación de placas de circuito impreso PCB, y el ISIS el cual permite simular circuitos con componentes desde resistencias, capacitores hasta integrados como microcontroladores, además de la comunicación serial y usb.

Para la simulación de la comunicación USB entre un computador y el Microcontrolador, ISIS requiere tener el USB Virtual instalado.

1.7.1.1. Instalación USB Virtual

- Clic en Inicio – Todos los Programas – Proteus 7 Professional – Virtual USB – Install USB Drivers

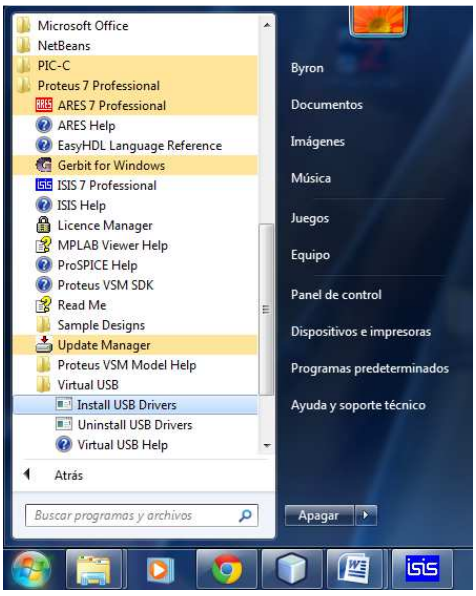


Figura 1. 7. Instalación de USB Drivers de Proteus

Autor: Byron Delpino

- Se instala de los drivers USB para Proteus que incluye el servicio de configuración y de limpieza (Cleanup).

Figura 1. 8. Instalación Componentes de USB Drivers

Autor: Byron Delpino

- La instalación de los controladores USB de Proteus finaliza, el PC emite el típico sonido indicando que un dispositivo está conectado. Se puede verificar que se ha instalado el puerto virtual en la ventana de Administración de Dispositivos, el puerto se denomina Eltima.

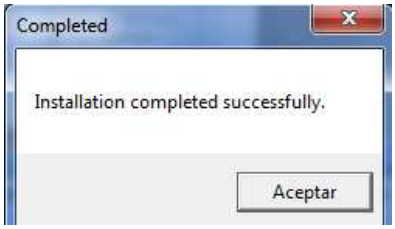


Figura 1. 9. Instalación Finalizada de USB Drivers de Proteus

Autor: Byron Delpino

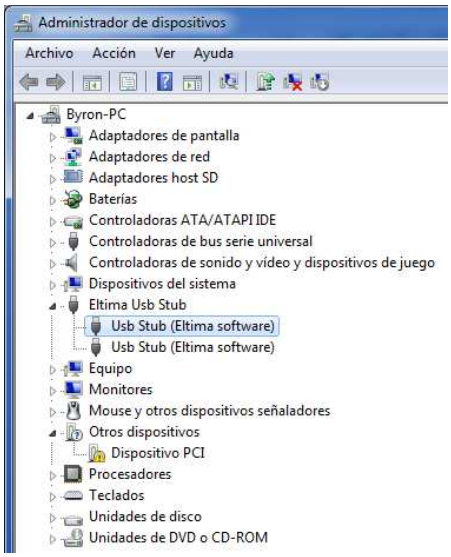


Figura 1. 10. Ventana de Administración de Dispositivos- USB Virtual Eltima

Autor: Byron Delpino

1.7.1.2 Diseño de Circuito en Proteus

El desarrollo del siguiente laboratorio consiste en la lectura de datos de un dipswitch.

Componentes a Utilizar:

- Microcontrolador PIC18F4550
- Conector USB USBCONN
- Dipswitch 8 entradas DIPSW_8
- Capacitor Electrolítico 47 uF CAP-ELEC
- Resistencias de 470Ω, 10KΩ
- VCC,GND

  Nota: Capacitor va conectado a VUSB para regularizar el voltaje, recomendado por el fabricante MICROCHIP.

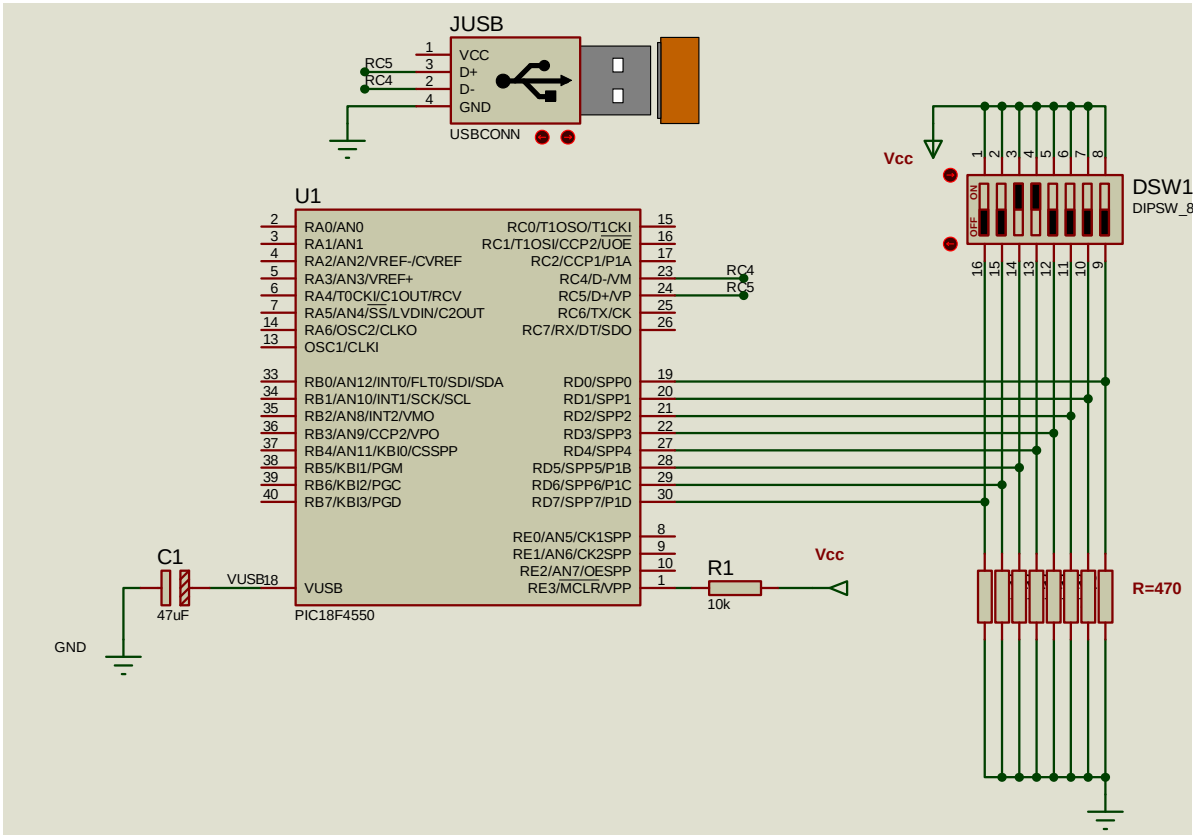


Figura 1. 11. Circuito diseñado en Isis, simulación lectura de datos USB

Autor: Byron Delpino

1.7.1.3 Diagrama de Bloques

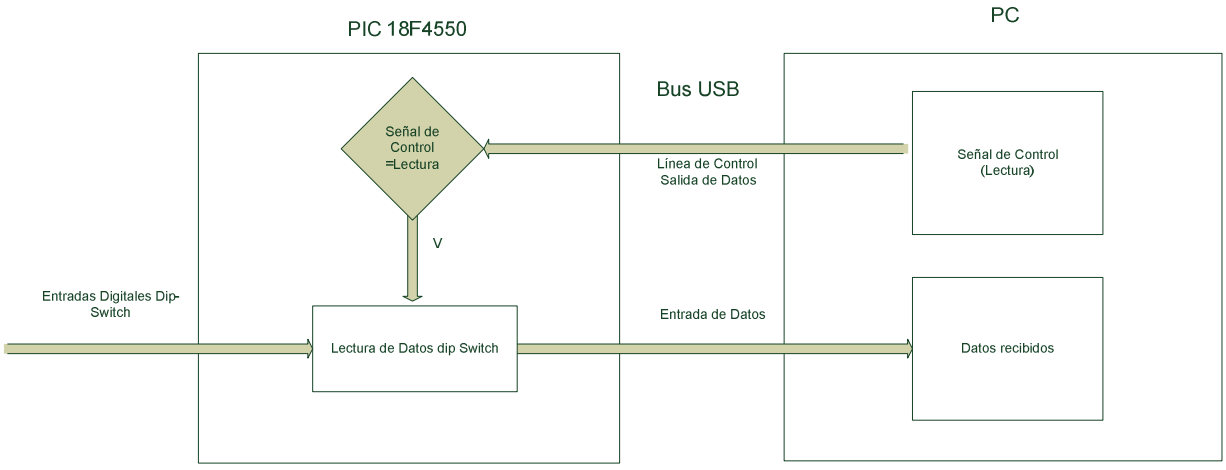


Figura 1. 12. Diagrama de Bloques, lectura de datos USB

Autor: Byron Delpino

1.7.1.4. Programación Java

- En Netbeans, crear una aplicación java, New File – Java Aplicaction, Nombre y Ubicación de la aplicación –Finish.
- En el proyecto se procede a crear un JFrame para diseñar la interfaz gráfica de la aplicación.

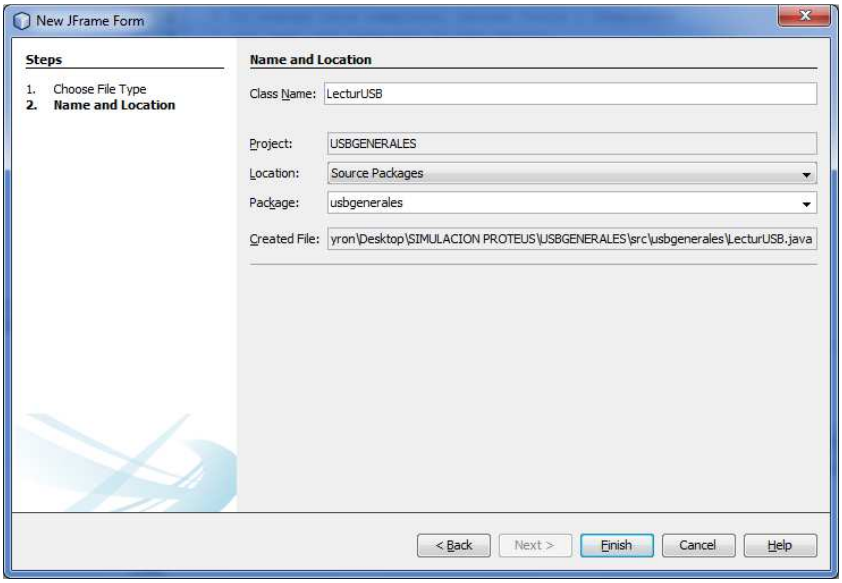


Figura 1. 13. Creación JFrame en Netbeans

Autor: Byron Delpino

- Se procede a realizar el diseño de la aplicación, insertando componentes como campos de texto, botones, etiquetas.

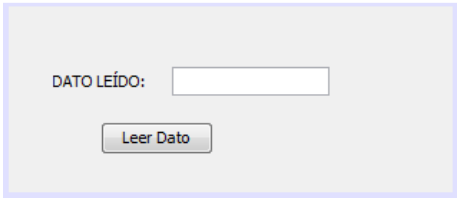


Figura 1. 14. Diseño de la Aplicación

Autor: Byron Delpino

- En la fuente del Jframe se ingresa el código de la aplicación, para la lectura datos de un dipswitch se utilizó métodos generales de la librería jpicusb.

1.7.1.4.1. Definición de Variables

Nombre	Tipo	Definición
VIDyPID	String	VID y PID del PIC
instancia	Entero	Instancia
lectura	Byte	Comando para Lectura de datos USB
out	Arreglo de Bytes	Envía el Comando a través del Puerto USB
dato	String	Almacena el dato recibido por el puerto USB

Tabla 1. 2. Definición de Variables de código java

Autor: Byron Delpino

1.7.1.5. Programación PIC C Compiler

1.7.1.5.1. Definición de Variables

Nombre	Tipo	Definición
recepcion	Arreglo de Enteros	Almacena la variable Comando
Comando	Entero	Variable que representa la lectura de datos
Digitos	Arreglo de char	Guarda la variable dato previamente convertida en arreglo de chars
dato	int	Recibe valor del puerto D del pic correspondiente al dip-switch

Tabla 1. 3. Definición de Variables de código C

Autor: Byron Delpino

1.7.1.6. Simulación Circuito

Una vez desarrollado el diseño y programación del laboratorio, se manda a simular dando clic en play. En un principio la simulación no se ejecutará debido a que no se ha instalado los drivers USB para el Microcontrolador 18F4550. Para esto se carga el driver Microchip para el Microcontrolador de esta forma la PC reconocerá e inicializará al dispositivo.

Para esto se debe seguir los siguientes pasos:

- Inicio – Clic Secundario en Equipo – Propiedades –Administrador de Dispositivos

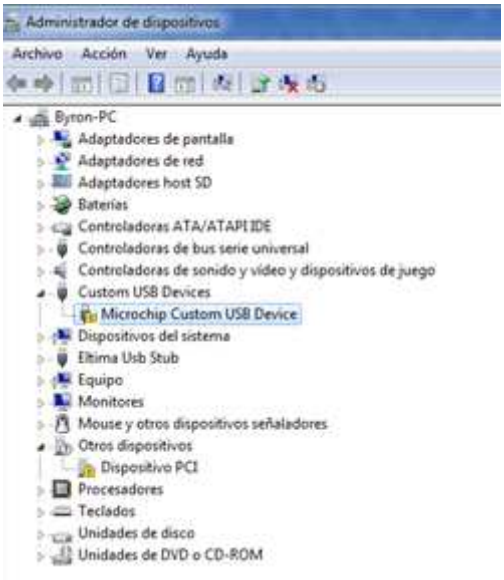


Figura 1. 15. Administrador de dispositivos- Dispositivo USB Microchip
Autor: Byron Delpino

- Clic derecho en el dispositivo desconocido – actualizar controlador. Se busca la carpeta donde se encuentra el controlador del dispositivo y se procede a instalar.
  Una vez que se instalado el controlador, el dispositivo estará listo para ser utilizado.

- Se procede a simular el circuito electrónico.

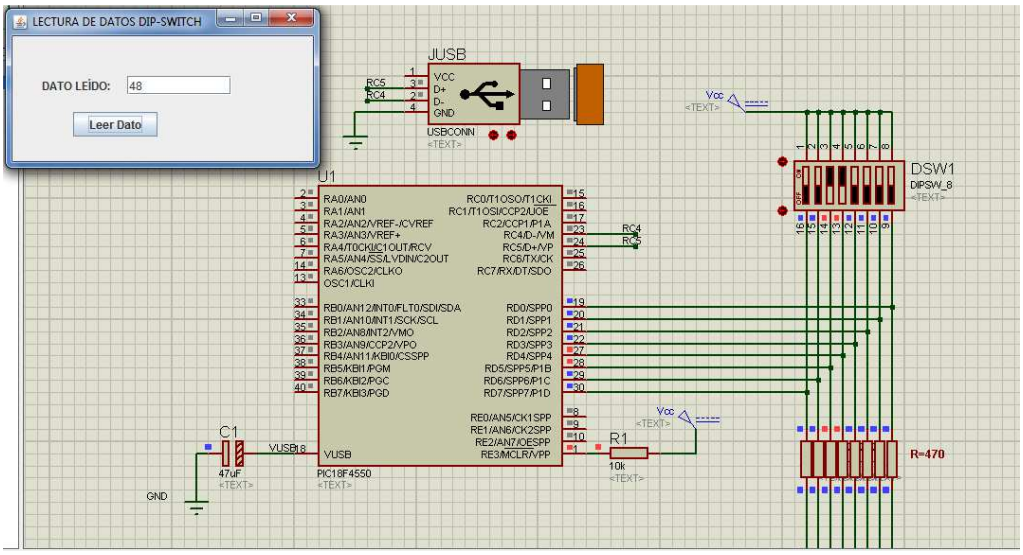


Figura 1. 16. Simulación lectura USB

Autor: Byron Delpino

1.7.2. Lectura y Escritura del puerto USB utilizando leds y dipwitch

Este laboratorio se centra en la escritura y lectura del puerto usb utilizando los métodos generales de la librería jpicusb.

1.7.2.1. Diseño de Circuito en Proteus

Componentes a Utilizar:

- Microcontrolador PIC18F4550
- Conector USB
- Dipswitch 8 entradas
- Leds
- Capacitor Electrolítico 47 uF
- Resistencias de 470Ω, 10KΩ
- Cristal 20Mhz
- Capacitores cerámicos 22pF
- Fuente de Alimentación

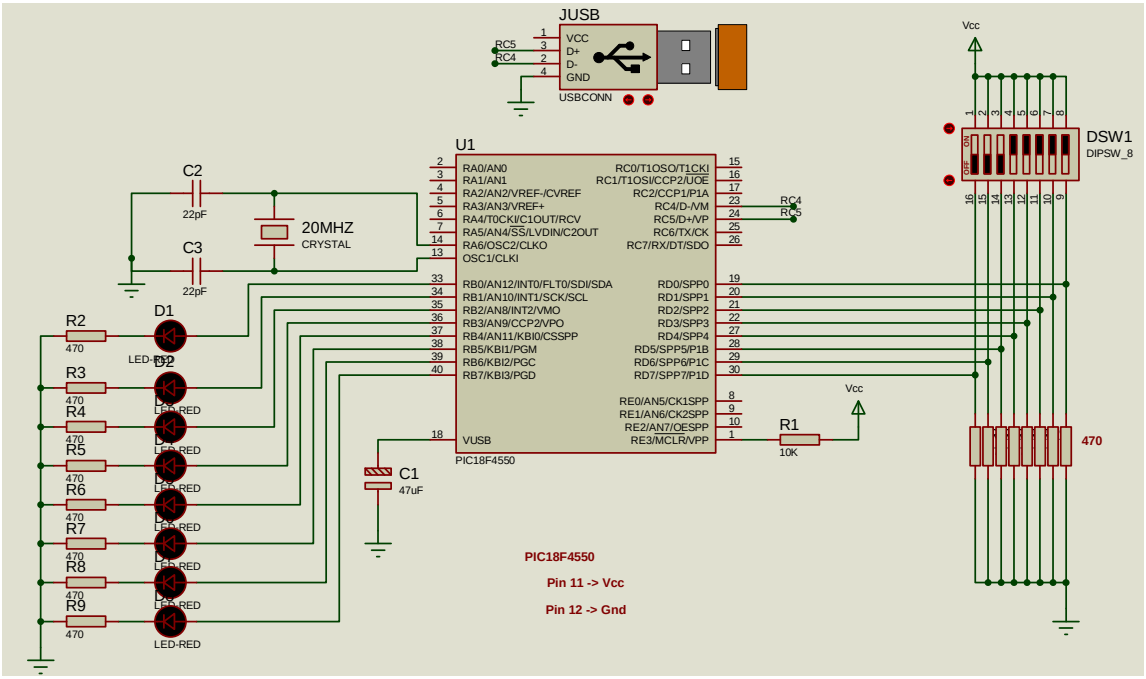


Figura 1. 17. Circuito diseñado en Isis, Lectura y Escritura del Puerto USB

Autor: Byron Delpino

1.7.2.2. Diagrama de Bloques

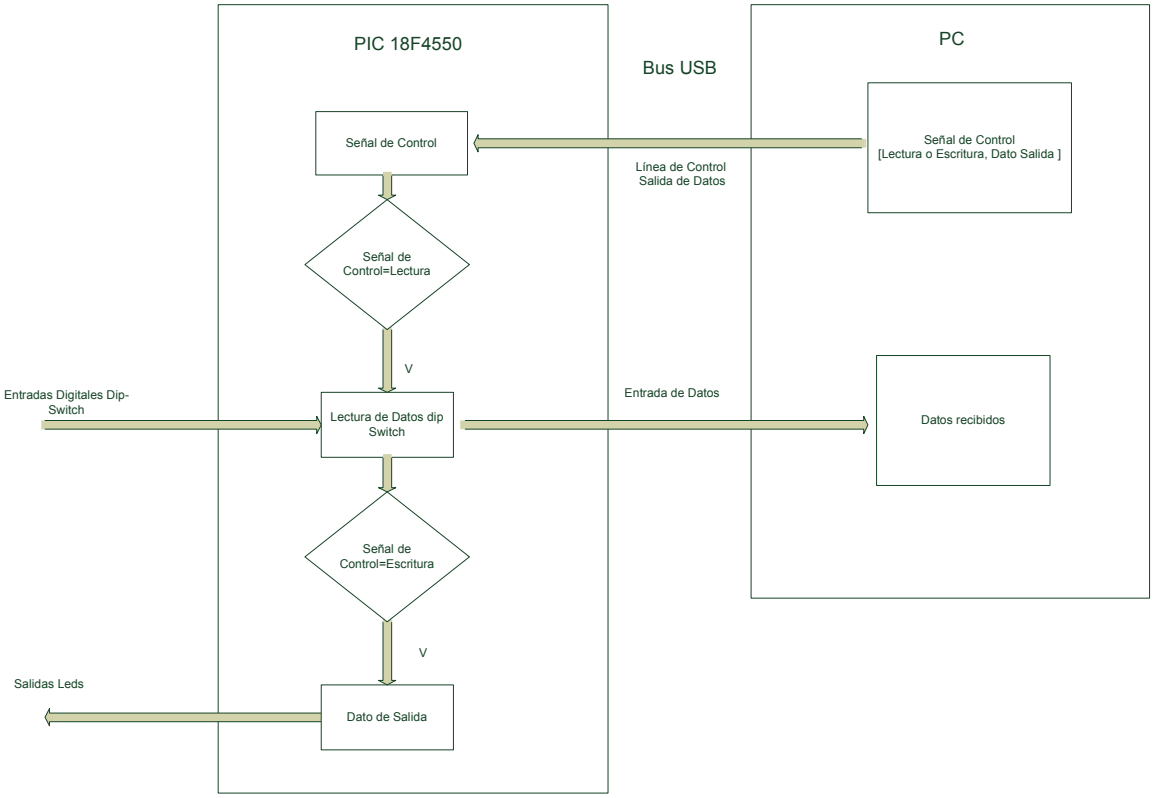


Figura 1. 18. Diagrama de Bloques, lectura y escritura de datos USB

Autor: Byron Delpino

1.7.2.3. Programación Java

- En Netbeans, crear una aplicación java, New File – Java Aplicaction, Nombre y Ubicación de la aplicación –Finish.

- En el proyecto se procede a crear un JFrame para diseñar la interfaz gráfica de la aplicación.

- Se procede a realizar el diseño de la aplicación, insertando componentes como campos de texto, check box, botones, etiquetas.

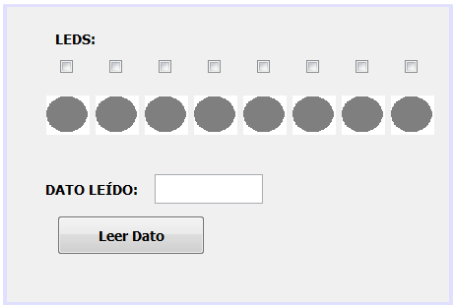


Figura 1. 19. Diseño de la Aplicación

Autor: Byron Delpino

- En la fuente del Jframe se ingresa el código de la aplicación, para la lectura datos de un dipswitch y encendido de leds.

1.7.2.3.1. Definición de Variables

Nombre	Tipo	Definición
VIDyPID	String	VID y PID del PIC
instancia	Entero	Instancia
lectura	Byte	Comando para Lectura de datos USB
escritura	Byte	Comando para Escritura de datos USB
out	Arreglo de Bytes	Envía la variable lectura o

		escritura a través del Puerto USB, además envía la variable DATO
DATO	Byte	Dato a ser enviado por el puerto USB
valor	String	Dato recibido desde puerto USB

Tabla 1. 4. Definición de Variables de código Java

Autor: Byron Delpino

1.7.2.4. Programación PIC C Compiler

1.7.2.4.1. Definición de Variables

Nombre	Tipo	Definición
recepcion	Arreglo de Enteros	Almacena la variable Comando/ValorLed
Comando	Entero	Variable que representa la lectura y escritura de datos
ValorLed	Entero	Variable a desplegar por el puerto B del pic, leds.
Digitos	Arreglo de char	Guarda la variable dato previamente convertida en arreglo de chars
dato	int	Recibe valor del puerto D del pic correspondiente al dip-switch

Tabla 1. 5. Definición de Variables de código C

Autor: Byron Delpino

1.7.3. Lectura y Escritura del puerto USB utilizando sensores y ventiladores

Este laboratorio se centra en la escritura y lectura del puerto usb utilizando los métodos rápidos de la librería jpicusb. En este laboratorio se incorpora hilos y canvas.

1.7.3.1. Diseño de Circuito en Proteus

Componentes a Utilizar:

- Microcontrolador PIC18F4550
- Conector USB
- Sensor de Temperatura LM35
- Ventilador 12V
- Transistor 3904
- Diodo 1N4007
- Capacitor Electrolítico 47 uF
- Resistencias de 220Ω, 10KΩ
- Cristal 20Mhz
- Capacitores cerámicos 22pF
- Fuente de Alimentación

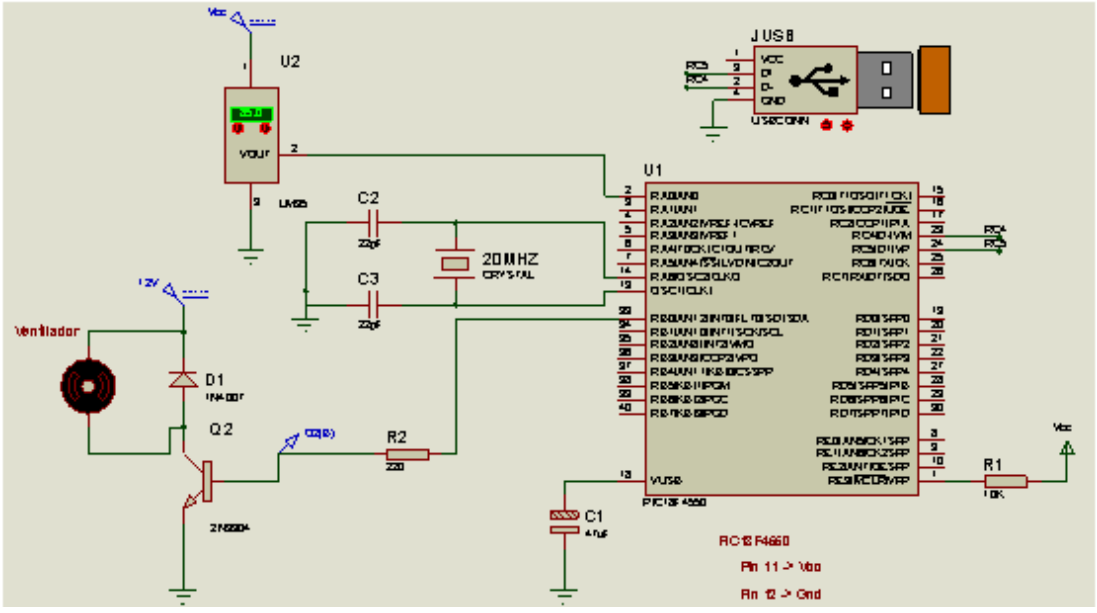


Figura 1. 20. Circuito diseñado en Isis, Lectura y Escritura del Puerto USB

Autor: Byron Delpino

1.7.3.2. Diagrama de Bloques

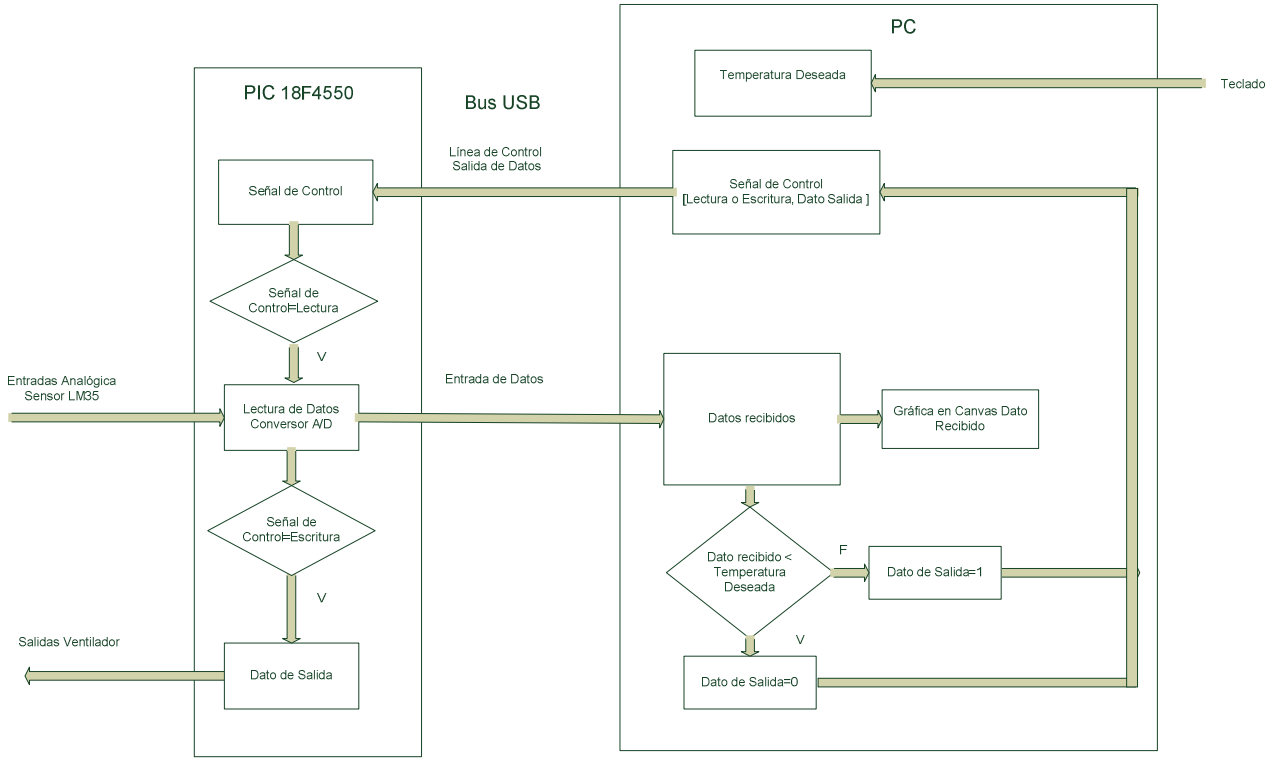


Figura 1. 21. Diagrama de Bloques, lectura y escritura de datos USB

Autor: Byron Delpino

1.7.3.3. Programación Java

- En Netbeans, crear una aplicación java, New File – Java Aplicaction, Nombre y Ubicación de la aplicación –Finish.

- En el proyecto se procede a crear un JFrame para diseñar la interfaz gráfica de la aplicación.

- Se procede a realizar el diseño de la aplicación, insertando componentes como campos de texto, check box, botones, etiquetas.

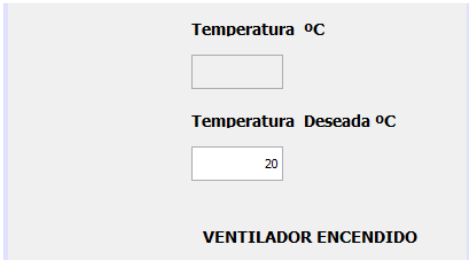


Figura 1. 22. Diseño de la Aplicación

Autor: Byron Delpino

1.7.3.3.1. Definición de Variables

Nombre	Tipo	Definición
VIDyPID	String	VID y PID del PIC
instancia	Entero	Instancia
lectura	Byte	Comando para Lectura de datos USB
escritura	Byte	Comando para Escritura de datos USB
out	Arreglo de Bytes	Envía la variable lectura o escritura a través del Puerto USB, además el valor a enviar por puerto USB
H	Hilo	Implementación de clase hilo
T	Thread	Clase Hilo hereda la clase Thread
dato	String	Dato recibido desde puerto USB
valseteo	String	Valor correspondiente a la temperatura deseada, se obtiene desde un campo de texto.
temperatura	Doble	Variable dato convertida en Doble
seteo	Doble	Variable valseteo convertida en Doble

Tabla 1. 6. Definición de Variables de código Java

Autor: Byron Delpino

1.7.3.4. Programación PIC Compiler

1.7.3.4.1. Definición de Variables

Nombre	Tipo	Definición
recepcion	Arreglo de Enteros	Almacena la variable Comando/Ventilador
Comando	Entero	Variable que representa la lectura y escritura de datos
Ventilador	Entero	Variable a desplegar por el puerto B del pic, ventilador
Digitos	Arreglo de char	Guarda la variable dato previamente convertida en arreglo de chars
a	int	Recibe valor del puerto A analógico del pic, LM35

Tabla 1. 7. Definición de Variables de código Java

Autor: Byron Delpino

JAVA SERVER FACES (JSF)

1.8. Conceptos Generales y fundamentación

El estudio de Java Server Faces involucra tener ciertos conocimientos previos que son de suma importancia para comprender esta tecnología, a continuación se detallan conceptos como la arquitectura cliente- servidor, protocolo HTTP, lenguajes html, jsp.

- **Arquitectura Cliente-Servidor**

Es un modelo de aplicación conformado por el cliente, que solicita un recurso o servicio y el servidor quien responde a dicha solicitud. Los servicios van desde correo electrónico, base de datos, web, archivos.

El cliente que a través de una interfaz gráfica interactúa con el usuario final inicia la comunicación con el servidor solicitándole un servicio.

Esta arquitectura facilita la distribución de tareas y funciones ya que permite distinguir los equipos que requieren de algún recurso y de aquellos que lo manejan, controlan y procesan.

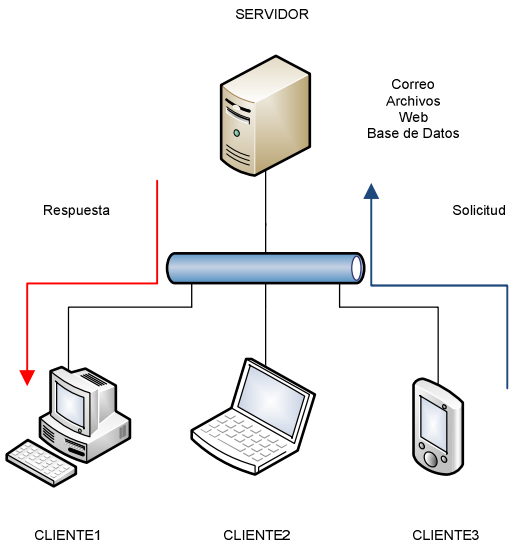


Figura 1. 23. Arquitectura Cliente-Servidor

Autor: Byron Delpino

- **HTTP (HyperText Transfer Protocol)**

HTTP (Protocolo de Transferencia de Hipertexto), es el protocolo utilizado en cada transacción www (World Wide Web), el cual define la sintaxis que utiliza el cliente y el servidor para comunicarse. La información o recurso transmitido se lo identifica mediante una URL (Localizador Uniforme de Recursos). El cliente realiza la solicitud a través de un navegador web o agente de usuario hacia un servidor que almacena y gestiona los recursos (Archivo, consulta de una base de datos, e/o).

La siguiente figura se observa como un cliente solicita al servidor de google su página web, este a su vez responde la solicitud enviando los recursos solicitados (logos, botones, campos de texto). La solicitud se da mediante la inserción de la URL **http://www.google.com** que es ingresada en un navegador web.

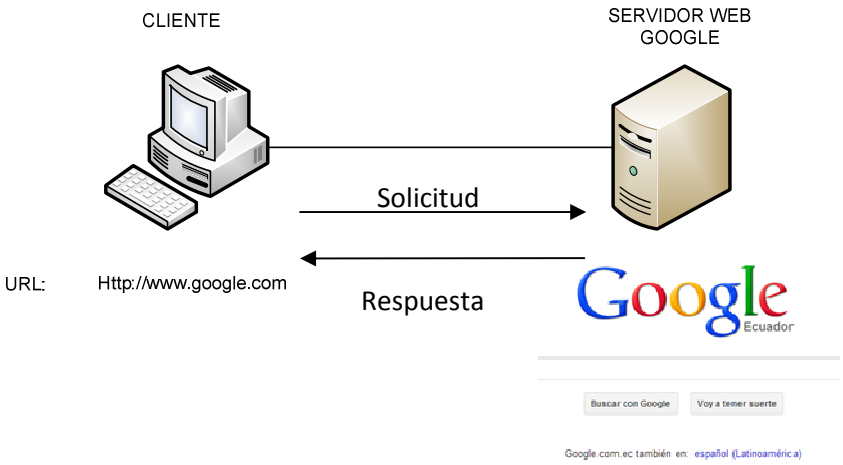


Figura 1. 24. Comunicación cliente-servidor web

Autor: Byron Delpino

- **HTML (HyperText Markup Language)**

HTML (Lenguaje Marcado de Hipertexto), es un lenguaje de programación utilizado para la elaboración de páginas web, HTML permite la inserción de texto, imágenes, recursos multimedia, además de la introducción de otros lenguajes como php, jsp, jsf. Una página html está conformada por una cabecera (**head),** que establece parámetros como el título de

la página, y el cuerpo (**body**), que representa la estructura interna de la misma a través de sus componentes (etiquetas, botones, imágenes).

```
<html>
<body>
<Titulo>
<p><strong>Ejemplo Pagina WEB html</strong></p>
<p>Escuela Politecnica del Ejercito</p>
<p> </p>
<Imagen>
<p><img src="http://www.ipgh.gob.ec/imagenes/logos/ESPE.gif" alt="espe"/></p>
</body>

</html>
```

Figura 1. 25. Código y Diseño de una página html

Autor: Byron Delpino

- **JSP (Java Server Pages)**

Es una tecnología JAVA desarrollada por Sun Microsystems que facilita la generación de contenido dinámico (interacción con el usuario, cálculos) en forma de documentos html.
JSP permite la utilización de código java mediante scripts, así como la inserción de etiquetas.

```
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <title>JSP Page</title>
    </head>
    <body>
        <h1>Escuela Politécnica del Ejército</h1>
        <h1>Ejemplo programación jsp</h1>
        <h1> Hoy es:<%= new java.util.Date() %>
</h1>
    </body>
</html>
```

Lenguaje JSP

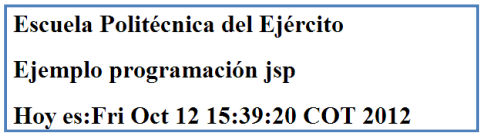


Figura 1. 26. Código y Diseño de una página con código JSP

Autor: Byron Delpino

- **Java Enterprise Edition (JEE)**

JEE, es una tecnología utilizada para desarrollar y ejecutar software de aplicaciones basado en programación java. Es una plataforma creada por Sun Microsystems en 1997 para el desarrollo de aplicaciones empresariales multicapa que se ejecutan sobre un servidor de aplicaciones.

La tecnología JEE presenta las siguientes características:

- ✓ Plataforma abierta y estándar: es posible crear aplicaciones web basadas única y exclusivamente en productos de software libre.

- ✓ Multiplataforma: Sus aplicaciones al ser desarrolladas en lenguaje java pueden ser ejecutadas en cualquier sistema operativo.

- ✓ Define un modelo de aplicaciones distribuido y multicapa de n niveles.

El diseño e implementación de aplicaciones web dentro de una arquitectura cliente-servidor conlleva la programación por capas. Estas capas están constituidas por:

- **Capa de Presentación:** Representa la interfaz gráfica del usuario**.**

- **Capa de Negocio:** Contiene los programas que se ejecutan, su tarea es la de recibir la solicitudes planteadas por el usuario y enviar las respuestas tras un proceso. Esta capa se comunica con la capa de presentación para recibir las solicitudes y presentar los resultados, y la de datos para solicitar el almacenamiento, recuperación de información dentro de un gestor de base de datos.

- **Capa de Datos:** Es donde residen los datos, es la encargada de acceder a los mismos.

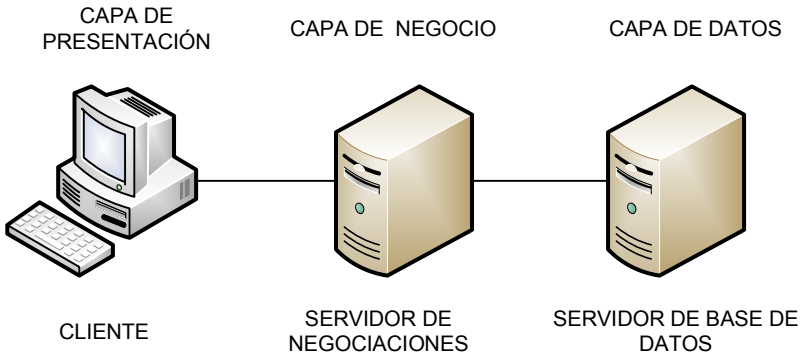


Figura 1. 27. Programación por Capas

Autor: Byron Delpino

La arquitectura JEE está definida en cuatro capas, la capa de cliente, la capa web, la capa de negocio y la capa de datos.

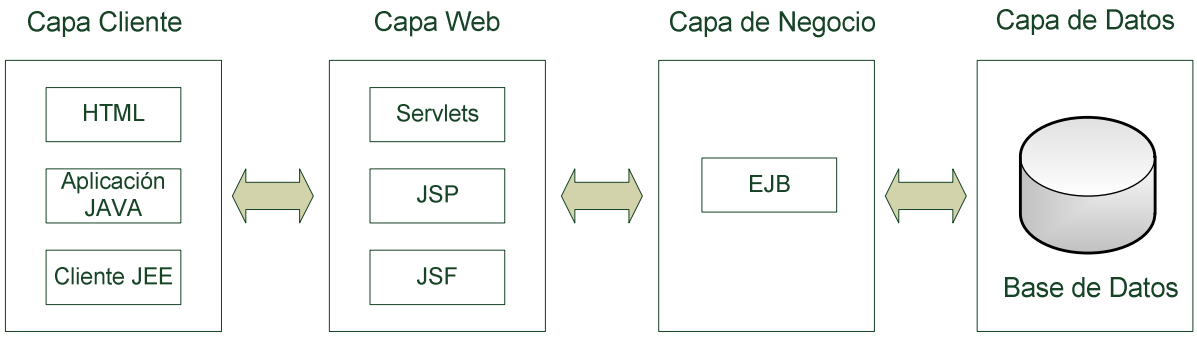


Figura 1. 28. Arquitectura JEE

Autor: Byron Delpino

- **Capa Cliente:** Es la interfaz gráfica que interactúa con el usuario. JEE soporta clientes HTML, applets, aplicaciones java, clientes Java Mobile.

- **Capa Web:** Es aquella donde se reciben los requerimientos del cliente, posee la lógica de presentación para generar una respuesta. En esta capa se tiene:

- ✓ Contenedores: Proveen un entorno de ejecución a los componentes web. Brindan servicios de seguridad, transacciones, comunicación. Ejm: JSP, Servlets.
- ✓ Frameworks: JSF, Struts.

- **Capa de Negocio:** Contiene la lógica de negocio (información propia) de la aplicación. Los componentes del negocio se comunican con la capa de datos. En el modelo JEE está representada por los EJB(Enterprise Java Beans).

- **Capa de Datos:** Representado por la base de datos o gestores de base de datos.

➢ **Servidor de Aplicaciones**

Para que los clientes accedan a las aplicaciones JEE requieren de un servidor de aplicaciones. Un servidor de aplicaciones es un software que proporciona servicios de aplicación al cliente, controlando su acceso y gestionando las funciones de lógica de negocio y datos. Existe un importante número de fabricantes de servidores de aplicaciones, algunos son gratuitos (Open Source), mientras otros son productos de pago. En la siguiente tabla se indica los principales servidores de aplicaciones y sus características:

Nombre	Propietario	Licencia	Multiplataforma	Compatibilidad con Apache Server	Implementación de EJB
WebLogic	Oracle	Si	Si	No	Si
Oracle Application Sever	Oracle	Si	Si	No	Si
JBoss	JBoss	No	Si	Si	Si
GlassFish	Oracle	No	Si	Si	Si
Tomcat	Apache	No	Si	Si	No
Genonimo	Apache	No	Si	Si	Si

Tabla 1. 8. Tabla Comparativa Servidores de Aplicaciones

Autor: Byron Delpino

- **GlassFish**

Es un servidor de aplicaciones JEE desarrollado por Sun Microsystems-Oracle. Este tipo de software (derivado de apache Tomcat) es gratuito. La última versión de GlassFish es la 3.1.2.

GlassFish es de fácil administración, provee servicios de seguridad en el desarrollo y desempeño de aplicaciones, escalabilidad, control de errores. GlassFish permite el ajuste y actualización de las aplicaciones web sin interrumpir su servicio.

GlassFish presenta las siguientes características:

- ✓ Menor tiempo de inicio del servidor.
- ✓ Bajo consumo de memoria RAM, reducción del empleo de recursos para el desarrollo de aplicaciones.
- ✓ Soporte para Netbeans y Eclipse.
- ✓ Configuración en varios idiomas.
- ✓ Servicio de Clustering (distribución de tareas en varios servidores- alta disponibilidad).
- ✓ Interoperabilidad con servidores web alojados en Windows.
- ✓ Integración con JBI(Java Business Integration) y EJB
- ✓ Incorpora tecnologías JSF, JSP, API para servicios web, JMS (Java Message Service).
- ✓ Compatibilidad con lenguajes script: Java Script, PHP.
- ✓ Posee la certificación Java, de esta forma obtiene acceso a los últimos avances de este lenguaje de programación.
- ✓ Integración con DTrace (herramienta que permite medir, controlar, registrar variables del sistema operativo).

Figura 1. 29. Logotipo servidor GlassFish

Fuente: www.glassfish.java.net/

- **JBoss**

Servidor de aplicaciones JEE gratuito, creado por JBoss Inc – Red Hat. La última versión de JBoss es la 7.1.1.

Sus características son:

- ✓ Mayor tiempo de inicio del servidor.
- ✓ Soporte para Netbeans y Eclipse.
- ✓ Servicio de Clustering (distribución de tareas en varios servidores- alta disponibilidad).
- ✓ Administración a través de JMX (Extensiones de administración Java)
- ✓ Servicios de middleware (comunicación con otros servicios: emails, archivos, directorios).
- ✓ Interoperabilidad con servidores web alojados en Windows.
- ✓ Integración con JBI(Java Business Integration) y EJB
- ✓ Incorpora tecnologías JSF, JSP, API para servicios web, JMS (Java Message Service).

Figura 1. 30. Logotipo Jboss. Inc

Fuente: www. ubuntubook.wordpress.com

Las aplicaciones JEE se desarrollan en cualquier servidor de aplicaciones sea JBoss o GlassFish, los dos presentan características de multiplataforma, clustering, integran tecnología EJB, JSF. GlassFish aventaja a JBoss en parámetros como el tiempo de inicio del servidor, facilidad en la administración, certificación Java.

- **Java Server Faces (JSF)**

Es una tecnología que constituye un marco de trabajo (framework) que permite la creación de aplicaciones web basadas en la tecnología Java JEE y en la arquitectura Modelo-Vista-Controlador MVC. JSF simplifica el desarrollo de interfaces de usuario en aplicaciones JEE.

JSF facilita la construcción de aplicaciones, proporcionando un entorno de trabajo vía web que gestiona las acciones producidas por el usuario en su página HTML y las traduce a eventos que son enviados al servidor con el objetivo de regenerar la página original y reflejar los cambios pertinentes provocados por dichas acciones.
JSF fue desarrollada por Sun Microsystems y Java Community Process en el año 2004, hasta el momento existen cinco versiones de esta tecnología:

- JSF 1.0 (11/03/2004): Lanzamiento inicial de las especificaciones de JSF.
- JSF 1.1 (27/05/2004): Solución de errores presentes en JSF 1.0.
- JSF 1.2 (11/05/2006): Versión con mejoras y solución de errores.
- JSF 2.0 (12/08/2009): Versión con mejoras de funcionalidad y facilidad de uso.
- JSF 2.1 (22-10-2010): Última versión, presenta mínimos cambios con respecto a JSF 2.0.

JSF presenta las siguientes características:

- Código JSF similar al código HTML estándar.
- Fácil manejo y manipulación.
- Posee un conjunto de API (Interfaz de Programación de Aplicaciones) para representar componentes de una interfaz de usuario (UI) que maneja su estado, eventos y navegación entre páginas.

- Utiliza páginas JSP para generar las vistas, añadiendo una biblioteca de etiquetas propia para crear los elementos de los formularios HTML.
- Resuelve validaciones, conversiones, mensajes de error e internacionalización (acceso a la aplicación en cualquier idioma).
- Al estar basado en tecnología JAVA, la programación de la interfaz gráfica se hace a través de componentes y manejo de eventos.
- Flexible: Permite crear cualquier tipo de componente y diseño.
- A diferencia de JSP, JSF ofrece una clara y total separación entre el comportamiento (manejo de eventos, transacciones) y presentación (interfaz gráfica).
- Es extensible, pudiendo crearse nuevos elementos de la interfaz o modificar los ya existentes

1.9. Modelo Vista Controlador (MVC)

MVC es un patrón o arquitectura de desarrollo de software que separa las capas de presentación, negocio y datos.

Este modelo de arquitectura presenta las siguientes ventajas:

- ✓ Separa los componentes de una aplicación, permitiendo implementar cada elemento por separado.
- ✓ Existencia de una API bien definida que facilita la separación del modelo, la vista y el controlador sin dificultad.
- ✓ Conexión dinámica entre el modelo y sus vistas.

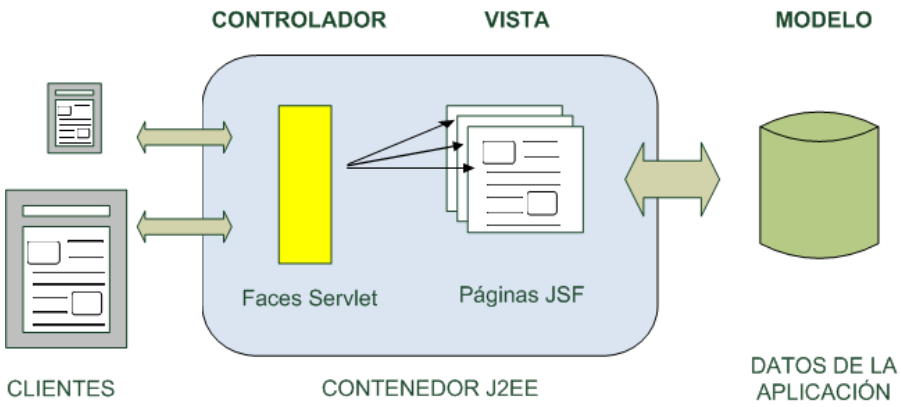


Figura 1. 31. Arquitectura MVC

Autor: Byron Delpino

Modelo: (Managed Beans): Objeto que se encarga de gestionar y controlar los datos del programa, respondiendo a los eventos generados por los componentes JSF. El modelo está representado por los Managed Beans.

Vista: **(clases xhtml, jsp, html)** Maneja la presentación visual de los datos gestionados por el modelo, usualmente es definida como la interfaz de usuario. Vincula los componentes JSF con los Managed Beans.

Controlador: Responde a eventos generados por el usuario, actuando sobre los datos representados por el modelo. Se encarga de acceder al modelo y actualizar la información así como generar una nueva vista. El controlador está representado por el Faces Servlet quien examina las peticiones recibidas, actualiza la interfaz del cliente y los datos del Managed Beans.

1.10. Ciclo de Vida JSF

El ciclo de vida de una aplicación JSF es similar a cualquier otro tipo de aplicación web donde el cliente hace una solicitud http y el servidor responde con una página html, sin embargo debido a la mayor prestación de servicios de JSF, se ejecuta algunos pasos adicionales.

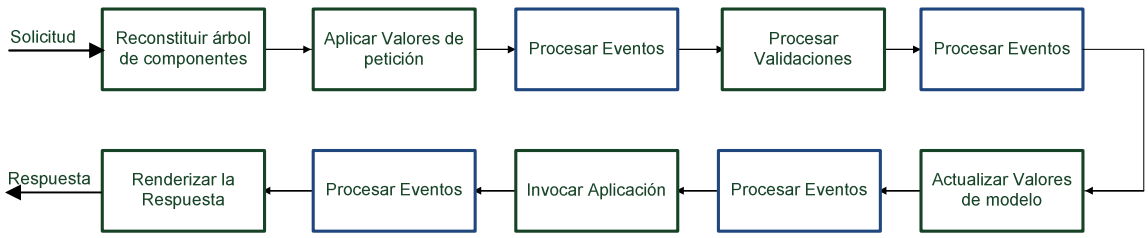


Figura 1. 32. Ciclo de Vida Aplicación JSF

Autor: Byron Delpino

✓ **Reconstruir el árbol de componentes o Restauración de la Vista**

Inicia cuando se realiza una petición HTTP o se produce un evento. Durante esta fase la implementación JSF construye el árbol con todos los componentes de la página JSF.

✓ **Aplicar valores de petición**

Cada componente del árbol creado en la fase anterior obtiene un valor desde los parámetros de la petición realizada y lo almacena.

✓ **Procesar validaciones**

Una vez que se almacena los valores de cada componente estos son validados de acuerdo a las reglas establecidas en la aplicación.

✓ **Actualizar los valores del modelo**

Los valores de los componentes son utilizados para actualizar el modelo por medio de los Beans.

✓ **Invocar la Aplicación**

Se ejecuta la acción correspondiente al evento que dio inicio al proceso. Ejm: Si se presionó un botón, esta fase se encarga de ejecutar una acción determinada**.**

✓ **Renderizar la respuesta**

La respuesta se renderiza y regresa al cliente. Se actualiza la vista.

- ✓ **Procesar eventos**

Representan la ejecución de cualquier evento producido durante el ciclo de vida JSF. Ejm: Si se en un inicio se ha pulsado un botón para enviar un formulario, pero inmediatamente se pulsa un enlace, JSF es capaz de procesar este nuevo evento.

1.11. Instalación y configuración del ambiente de desarrollo

Instalación de Servidor GlassFish sobre Netbeans

- ✓ Arrancar Netbeans
- ✓ Dar clic en la pestaña **Services,** ubicada en la parte superior izquierda.

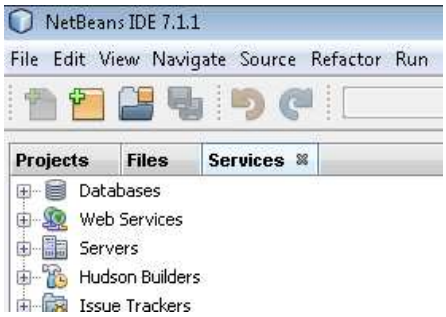


Figura 1. 33. Pestaña de Servicios Netbeans

Autor: Byron Delpino

- ✓ Clic derecho sobre **Servers** y clic sobre **Add Server**.

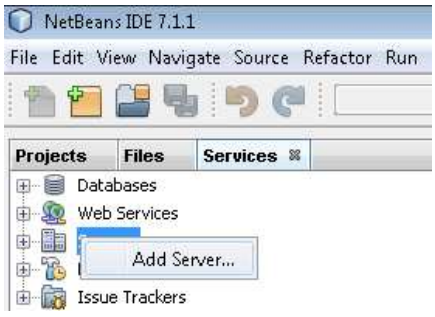


Figura 1. 34. Agregar Nuevo Servidor

Autor: Byron Delpino

✓ A continuación se despliega una ventana que permite seleccionar el servidor de aplicaciones deseado, se escoge **GlassFish Server 3+** y se da clic en **Next**.

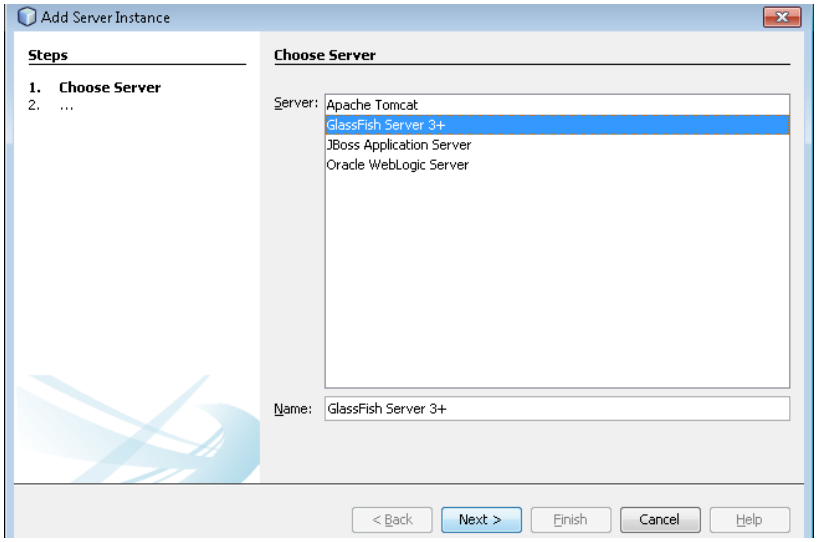


Figura 1. 35. Ventana de Selección de Nuevo Servidor de Aplicaciones

Autor: Byron Delpino

✓ Se procede a determinar la ubicación donde se descargará el servidor, se aceptan las condiciones y términos de licencia, y se da clic sobre el botón **Download Now**.

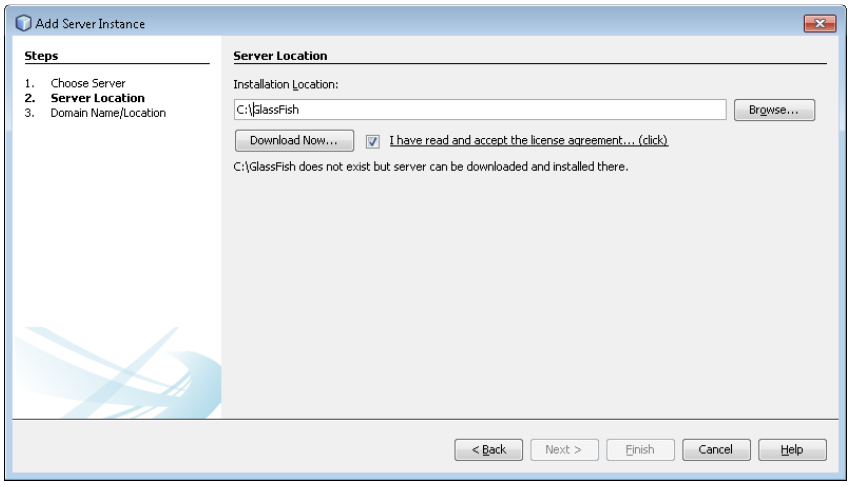


Figura 1. 36. Ventana de Selección de la Ubicación

Autor: Byron Delpino

- ✓ Se elige la versión de GlassFish, clic en el botón **Aceptar**.

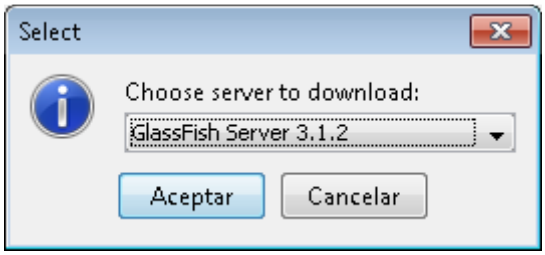


Figura 1. 37. Selección de Versión de GlassFish a instalar

Autor: Byron Delpino

- ✓ Una vez descargado el servidor en el host se da clic en **Next**.
- ✓ Se ingresa el nombre del dominio local, así como el puerto (8080 por defecto). Clic en **Finish**.

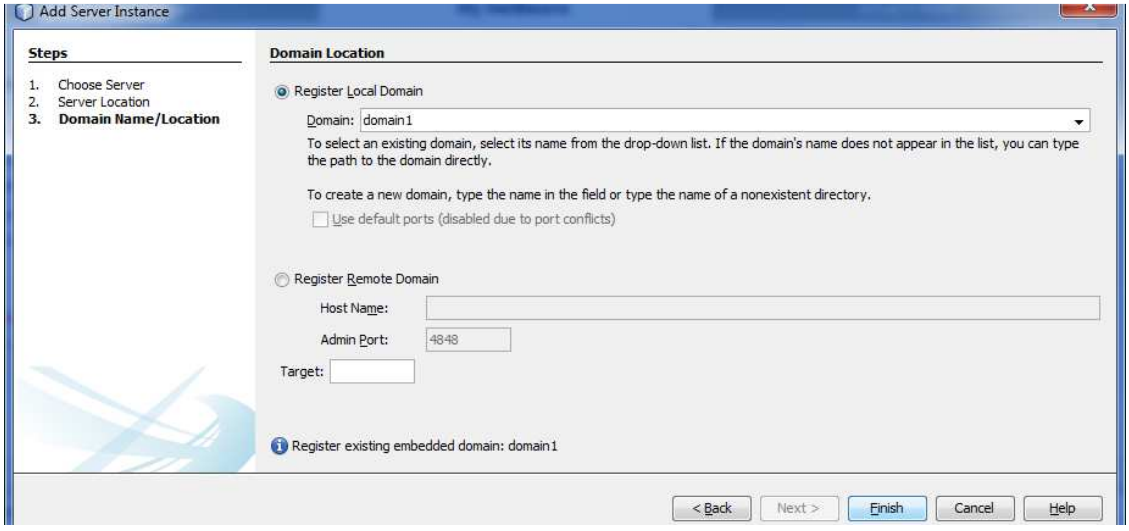


Figura 1. 38. Registro de dominio local y puerto de administración.

Autor: Byron Delpino

- ✓ Se arranca el servidor GlassFish, en la pestaña **Servicios, Server,** clic derecho **sobre GlassFish** y **Start.**

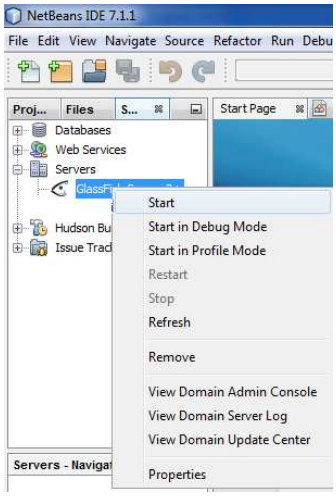


Figura 1. 39. Arranque del Servidor de GlassFish

Autor: Byron Delpino

Instalación de Servidor JBoss 7.1 sobre Eclipse

- ✓ Se descarga la versión 7.1.1 de JBoss desde la página de Jboss.Inc. http://www.jboss.org/jbossas/downloads/. Se descomprime el archivo.
- ✓ Arrancar Eclipse
- ✓ En un inicio se debe instalar la herramientas de JBoss, las cuales permitirán instalar el servidor JBoss 7.1. Clic en la pestaña **Window, Preferences**.
- ✓ En la ventana de **Preferences**, clic en **Server** => **Runtime Environments** => **Add.**

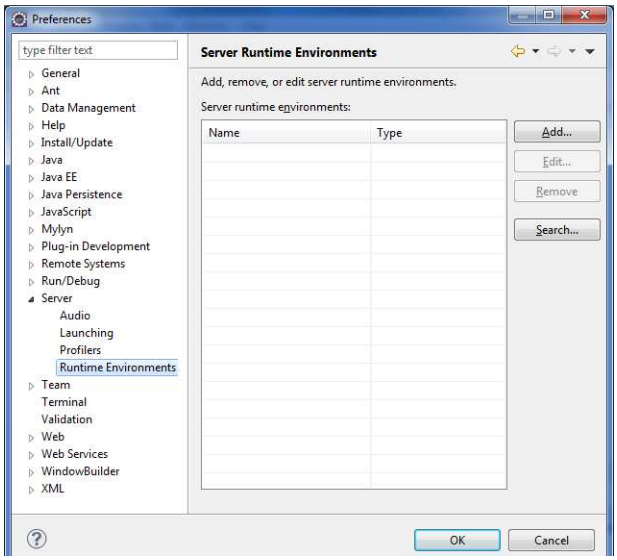


Figura 1. 40. Adición de un nuevo servidor de aplicaciones

Autor: Byron Delpino

✓ Clic en **Download additional server adapters**.

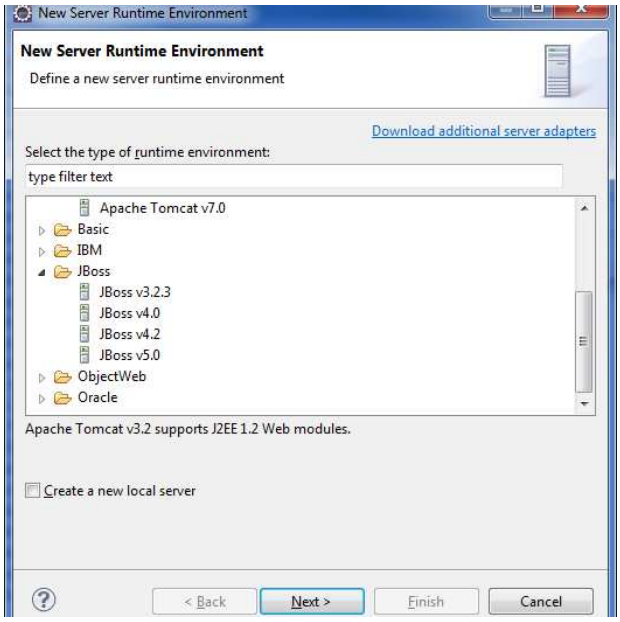


Figura 1. 41. Ventana de Selección de un servidor de aplicaciones

Autor: Byron Delpino

✓ Se selecciona **JBossAS Tools**, clic **Next**.

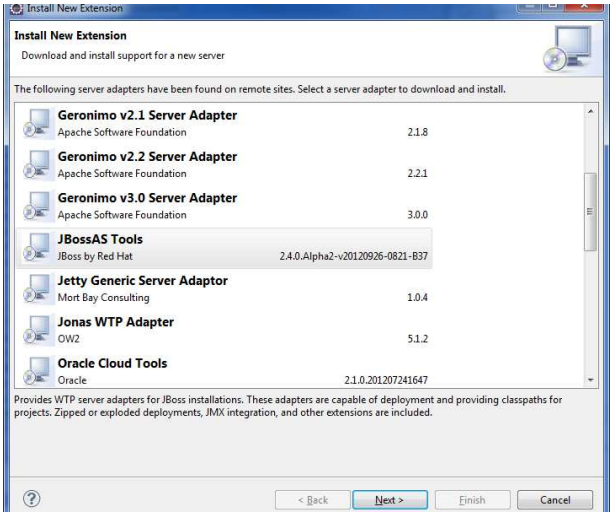


Figura 1. 42. Ventana de Selección de la herramienta JBossAS Tools

Autor: Byron Delpino

✓ Se acepta los términos y condiciones de licencia, clic en **Finish.**

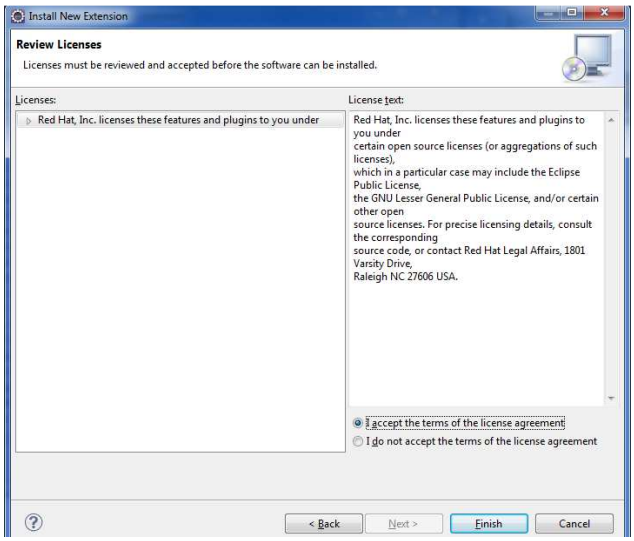


Figura 1. 43. Términos y condiciones de Licencia de JBossAS Tools

Autor: Byron Delpino

✓ Se instala el paquete JBossAS Tools, y se procede a reiniciar el Eclipse

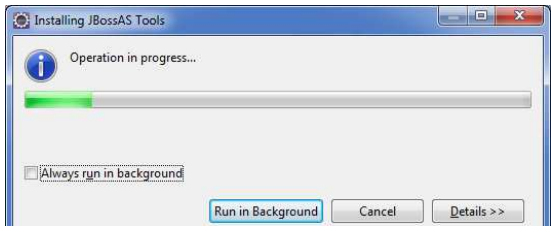


Figura 1. 44. Proceso de Instalación de JBossAS Tools

Autor: Byron Delpino

✓ Una vez que se ha instalado esta herramienta, se continua con la instalación de servidor de JBoss, clic en **File => New => Other => Server =>Next**.

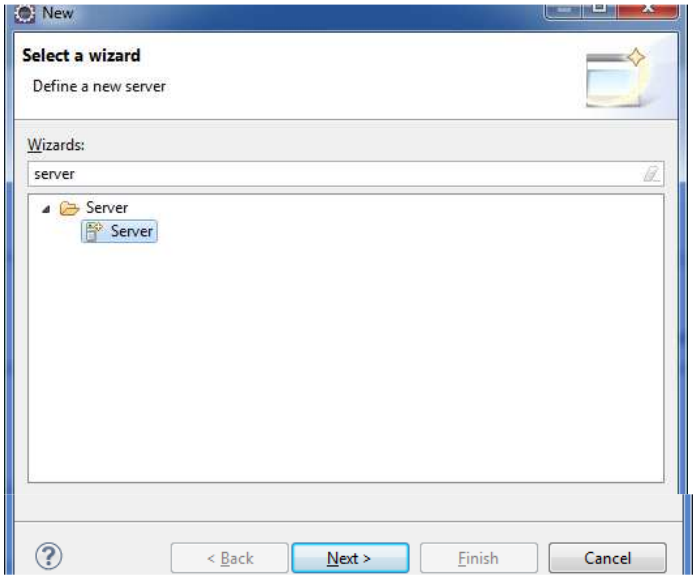


Figura 1. 45. Instalación de un nuevo servidor

Autor: Byron Delpino

✓ Se escoge **JBoss AS 7.1**, clic en **Next**.

Figura 1. 46. Instalación del Servidor JBoss 7.1

Autor: Byron Delpino

✓ Se determina el directorio donde se descomprimió el servidor descargado. Clic en **Finish.**

Figura 1. 47. Definición del directorio Servidor JBoss 7.1

Autor: Byron Delpino

- ✓ Clic en **Finish** para completar la instalación del servidor de aplicaciones.

- ✓ Se arranca JBoss dando clic en **Servers,** clic derecho sobre JBoss y **Start**.

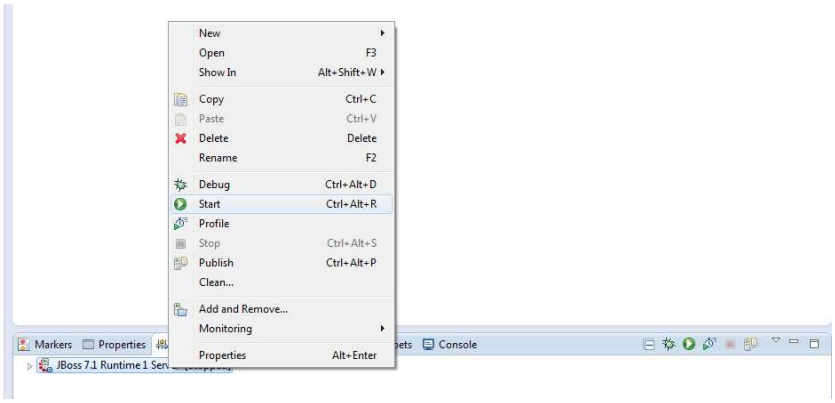


Figura 1. 48. Arranque del Servidor JBoss 7.1

Autor: Byron Delpino

1.12. Estructura básica de una aplicación JSF

Una aplicación Java Server Faces al igual que la mayoría de aplicaciones web se ejecuta por medio de un servidor web. Una aplicación JSF está constituida por:

- ✓ Componentes JavaBeans: Objetos que contienen los datos y funcionalidades especificas de la aplicación.
- ✓ Oyentes de Eventos.
- ✓ Páginas: Cada página JSF está formada por una página JSP que contiene un formulario HTML.
- ✓ Beans: Clases que permiten acceder a la base de datos
- ✓ Renderizadores: Asignan y calculan todos los códigos y propiedades JSF para ser mostrados en un formulario HTML.
- ✓ Convertidores: Transforman el valor de los componentes JSF (string) a objetos java (Entero, Double, Date) y viceversa.
- ✓ Validadores: Comprueban que los valores recibidos y almacenados en los componentes JSF cumplan las restricciones especificadas.

La estructura básica de una aplicación JSF está conformada por:

- ✓ **Páginas (xhtml):** Representa la interfaz gráfica del usuario.

- ✓ **Beans:** Clase Java que maneja y gestiona los datos del usuario.

- ✓ **Archivos de Configuración (xml):** Representa el controlador de la aplicación JSF.
  - o **face-config.xml:** Invoca a los Beans e implementa reglas de navegación, que permitirán enlazar a la vista (páginas) con el modelo (Beans).
  - o **web.xml:** Archivo que contiene información sobre las características de la aplicación (nombre, servlet, páginas de introducción).

- ✓ **Librerías**: Almacena el jdk, así como el servidor de aplicaciones que ejecutará la aplicación.

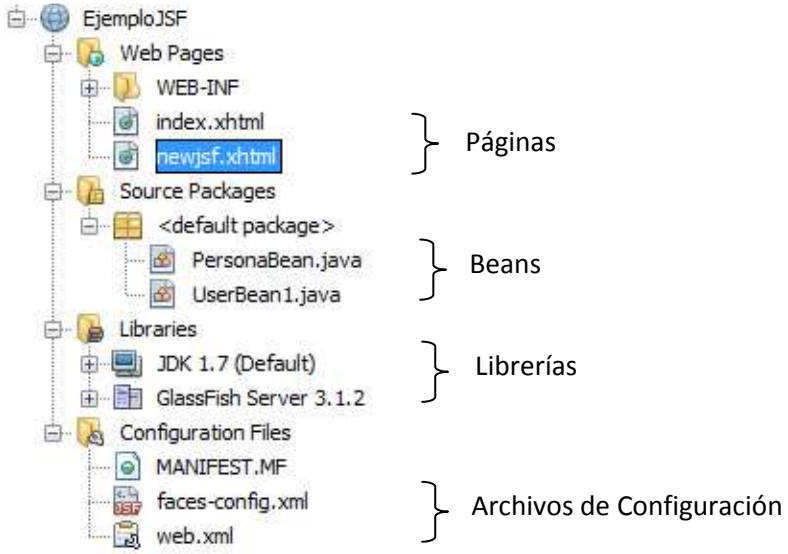


Figura 1. 49. Estructura de una Aplicación JSF

Autor: Byron Delpino

1.13. Creación y uso de Managed Beans

Un bean es una clase java que tiene la capacidad de ser reutilizable, permitiendo la separación entre la vista y el controlador.

Los Beans son utilizados cuando se requiere conectar a las páginas web con clases java.

Las características de un bean son sus atributos, cada atributo posee un nombre, tipo, y métodos para establecer (setter) u obtener (getter) el valor de cada atributo.

Para obtener el valor de un atributo de nombre xyz, se emplea el método getter a través de la sintaxis **getXyz**.

Para establecer el valor de una campo o atributo xyx, se utiliza el método setter **setXyz**.

Los métodos get y set son obligatorios para definir los valores de los atributos, aparte de estos se podrán incluir métodos adicionales que serán invocados a partir de un evento (Ejm: Dar clic sobre un botón).

Ejemplo:

Se tiene un Bean llamado PERSONA, este posee los atributos Cédula, Nombre y Edad de tipos String y Entero. En el bean se deberán definir los métodos que permitan obtener o establecer el valor de cada atributo.

BEAN: **PERSONA**		
Nombre Atributo	**Tipo**	**Métodos**
cedula	String	getCedula/setCedula
nombre	String	getNombre/setNombre
edad	Entero	getEdad/setEdad

Tabla 1. 9. Ejemplo propiedades de los atributos de un Bean

Autor: Byron Delpino

Además se tiene el método **validar**, utilizado para validar el atributo edad. Este método será utilizado cuando se genere un evento.

```
package Bean;

// Definición del Bean usuario
public class Persona {

    private String cedula, nombre;  // variables que almacenan cedula y nombre
    int edad; // variable que almacena edad


    //MÉTODOS OBLIGATORIOS EN EL BEAN

    //get y set para cada atributo

    public String getCedula() {  // método para obtener valor del atributo cedula
        return cedula;
    }

    public void setCedula(String Cedula) { //Metodo para establecer el valor del atributo nombre
        cedula=Cedula;
    }


    public String getNombre() {  // método para obtener valor del atributo cedula
        return nombre;
    }

    public void setNombre(String Nombre) { //Metodo para establecer el valor del atributo nombre
        nombre = Nombre;



    public String getEdad() {  // método para obtener valor del atributo edad
        return String.valueOf(edad); // retorno de valor edad convertido a String
    }

    public void setEdad(String Edad) { //Metodo para establecer el valor del atributo edad
        edad = Integer.parseInt(Edad); // Se convierte Edad a entero y se toma su valor
    }



    // MÉTODOS ADICIONALES
    //invocados a través de un avento

    public String validar(){
        if(edad <18)
            return "Mayor";
        else
            return "Menor";
    }
}
```

Definición de Variables (atributos)

Métodos get y set . **Obligatorios**

Métodos Adicionales

1.13.1. Ámbitos de Beans

Ámbito de tipo Petición

Es un ámbito basado en el modelo petición-respuesta. El cliente solicita algún recurso y el servidor responde, sin almacenar ninguna información.

Ejemplo:

Una aplicación web imprime una variable=0, se tiene un botón que al dar clic sobre este invoca a un método que imprime el valor de dicha variable y aumenta su valor en 1, si el da clic sobre el botón, el valor de la variable será 1, si de nuevo presiona el botón el valor será 1, esto se debe a que en el ámbito request no se almacena ningún tipo de información.

Figura 1. 50. Ejemplo de Ámbito Petición

Autor: Byron Delpino

Ámbito de tipo Sesión

Basado en el modelo petición-respuesta, el cliente solicita algún recurso al servidor y este le responde, la principal diferencia radica en el uso de cookies que permiten almacenar cierta información del cliente, así como las variables de la aplicación, la información guardada solo se da mientras dure la sesión del cliente.

Ejemplo:

Un cliente **A** ejecuta una aplicación web que imprime una variable=0, se tiene un botón que al dar clic sobre este invoca a un método que imprime el valor de dicha variable y aumenta su valor en 1, si **A** presiona el botón, el valor de la variable será 1, si de nuevo da clic sobre el botón la variable será 2, debido a que el ámbito sesión si almacena información del usuario y las variables.

Figura 1. 51. Ejemplo de Ámbito Sesión desde un cliente A

Autor: Byron Delpino

Si un cliente **B** ejecuta la aplicación web el valor de la variable será 0, puesto que corresponde a otra sesión.

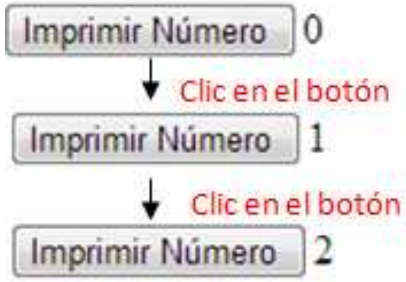


Figura 1. 52. Ejemplo de Ámbito Sesión desde un cliente B

Autor: Byron Delpino

Ámbito de tipo Aplicación

Basado en el modelo petición-respuesta, el cliente solicita algún recurso al servidor y este le responde, guardando información de las variables de la aplicación sin importar la sesión o el cliente que la solicite.

Ejemplo:

Un cliente **A** ejecuta una aplicación web que imprime una variable=0, se tiene un botón que al dar clic sobre este invoca a un método que imprime el valor de dicha variable y aumenta su valor en 1, si **A** presiona el botón, el valor de la variable será 1, si de nuevo da clic sobre el botón la variable será 2.

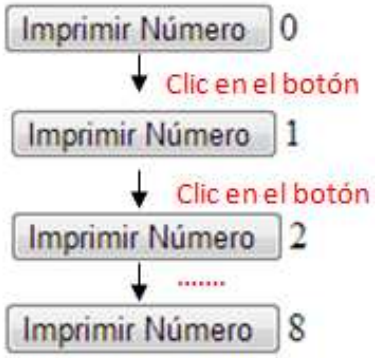


Figura 1. 53. Ejemplo de Ámbito Aplicación desde un cliente A

Autor: Byron Delpino

Si un cliente **B** ejecuta la aplicación el valor de la variable será 8 puesto que el ámbito tipo aplicación como su nombre lo indica se basa en la aplicación y no en la sesión de un determinado cliente.

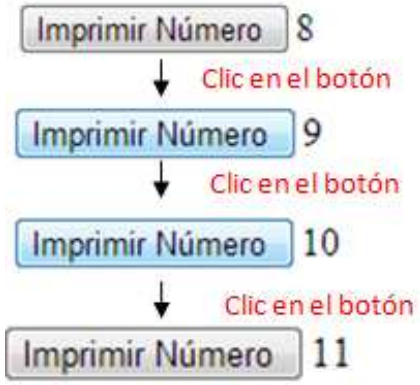


Figura 1. 54. Ejemplo de Ámbito Aplicación desde un cliente B

Autor: Byron Delpino

1.13.2. Configuración de Beans

El archivo faces-config.xml es el utilizado para configurar las características de los Beans, así como de los atributos que lo conforman.

Los parámetros a especificar en un bean incluyen:

- ✓ Nombre del bean: **managed-bean-name**

- ✓ Clase que contiene el bean: **managed-bean-class**
- ✓ Ámbito del Bean (none, aplication, session, request): **managed-bean-scope**
- ✓ Propiedades del Bean (atributos): **managed-property**
  - o Nombre de atributo o propiedad: **property-name**
  - o Tipos de Valores del atributo (String,Integer,Double,Long): **value-class**
  - o Inicialización de valores de atributo: **value/null-value**
  - o Uso de listas para ingresar varios valores al atributo: **list-entries**

Ejemplo:

Nombre del Bean: usuario **Ubicación**: Paquete.UsuarioBean		
Nombre Atributo	**Tipo**	**Valor inicial**
nombre	String	Byron
clave	String	Nulo
calificaciones	Doble	{18.5 20 10 18 0.5 }

Tabla 1. 10. Ejemplo de parámetros de configuración Bean usuario

Autor: Byron Delpino

```
<managed-bean>
    <managed-bean-name>usuario</managed-bean-name>
    <managed-bean-class>Paquete.UsuarioBean</managed-bean-class>
    <managed-bean-scope>request</managed-bean-scope >

    <managed-property>
        <property-name>nombre</property-name>
        <value>Byron</value>
    </managed-property>

    <managed-property>
        <property-name>clave</property-name>
        <null-value />
    </managed-property>

    <managed-property>
        <property-name>calificaciones</property-name>
        <list-entries>
            <value-class>java.lang.Double</value-class>
            <value>18.5</value>
            <value>20</value>
            <value>10</value>
            <value>18</value>
            <value>0.5</value>
        </list-entries>
    </managed-property>

</managed-bean>
```

1.13.3. Navegación

En la arquitectura Modelo-Vista-Controlador (MVC), el controlador es el encargado de implementar reglas de navegación, es decir especifica los mecanismos para navegar de una página a otras.

Existen dos tipos de navegación:

Navegación Estática

En este tipo de navegación, el usuario navega dependiendo de los botones o enlaces que presiona.

Ejemplo: Un usuario, se encuentra en la página de la Espe, si este da clic sobre el enlace **MI ESPE**, el navegador se dirigirá al login del portal MI ESPE.

a) Portal web de la Espe

b) Login del Portal MI ESPE

Figura 1. 55. Ejemplo de Navegación Estática

Autor: Byron Delpino

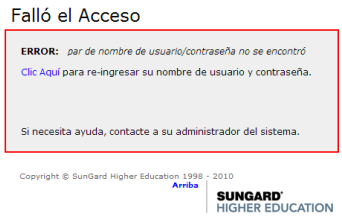
Navegación Dinámica

En la navegación dinámica, el flujo de páginas depende del botón que se pulsa, así como los datos ingresados.

Ejemplo: Si el usuario se encuentra en el login del portal MI ESPE, para ingresar al portal, deberá ingresar los datos correspondientes a **Usuario** y **Clave**, y dar clic en **Ingresar,** en el momento que ha presionado Ingresar, los datos son validados, si estos son correctos el navegador se dirige al portal, caso contrario el navegador se direccionará hacia otra página, indicando mensaje de error.

a) Login del Portal MI ESPE

b) Portal MI ESPE



c) Página que indica error en el ingreso de datos de usuario

Figura 1. 56. Ejemplo de Navegación Dinámica

Autor: Byron Delpino

En el archivo faces-config.xml al igual que la configuración de los bean, se establece las reglas de navegación.

Los parámetros a especificar en una regla de navegación incluyen:

- ✓ Página de Origen: **from-view-id**
- ✓ Página de Destino: **to-view-id**
- ✓ Posibles rutas o páginas de navegación: **navigation-case**
  - o Identificador o elemento retornado tras una acción: **from-outcome,** el cual servirá para direccionar hacia la página de destino.

Ejemplo: Si se presiona un botón que invoque a un método que valide los datos ingresados de usuario y contraseña, el método retorna un String "ingrese" o "Error", que depende si los datos insertados son correctos o no, este valor será utilizado como un identificador en el navegador a través del comando **from-outcome,** el cual permitirá la navegación de Login.xhtml a Usuario.xhtml o Error.xhtml.

```
<navigation-rule>
    <from-view-id>/Login.xhtml</from-view-id>
    <navigation-case>
        <from-outcome>ingrese</from-outcome>
        <to-view-id>/Usuario.xhtml</to-view-id>
    </navigation-case>
    <navigation-case>
        <from-outcome>error</from-outcome>
        <to-view-id>/Error.xhtml</to-view-id>
    </navigation-case>
</navigation-rule>
```

1.14. Etiquetas JSF

La tecnología JSF implementa dos tipos de etiquetas: Las **core,** utilizadas para el manejo de eventos, conversiones, validaciones, y las **html,** que se usan para la construcción de formularios e implementación de la interfaz de usuario.

Para el uso de etiquetas **core** se define el prefijo **f** (xmlns:f="http://java.sun.com/jsf/core"), mientras para las etiquetas **html**, se usa el prefijo **h (**xmlns:h="http://java.sun.com/jsf/html").

1.14.1. Etiquetas Core

Son etiquetas JSF utilizadas para el despliegue de vistas, subvistas, el uso de validaciones en el ingreso de datos, conversiones, así como el despliegue de ítems en el manejo de componentes de selección.

Nombre Etiqueta	Función
view	Crea una vista
subview	Crea una subvista
facet	Añade una faceta
attribute	Añade un atributo a un componente
param	Añade un parámetro a un componente
actionListener	Añade una acción a un componente
valueChangeListener	Añade un nuevo valor al oyente de un componente
convertDateTime	Añade una conversión de fecha a un componente
convertNumber	Añade una conversión de número a un componente
Validator	Añade un validador a un componente
validateLength	Valida la longitud del valor de un componente
validateDoubleRange	Valida un rango tipo doublé a un componente
validateLongRange	Valida un rango de tipo long para valores de componentes
loadBundle	Carga el origen de un elemento Bundle
selectitems	Especifica los elementos para seleccionar uno o varios elementos
selectitem	Especifica un elemento para seleccionar uno o varios elementos
verbaitim	Añade una marca a una página jsf

Tabla 1. 11. Etiquetas Core de la Tecnología JSF

Autor: Byron Delpino

1.14.2. Etiquetas Html

Componentes JSF que representan la interfaz gráfica (vista) a través de campos de texto, etiquetas, imágenes, botones, formularios.

Nombre Etiqueta	Función
form	Formulario html
inputText	Campo de Texto de entrada
inputTextarea	Área de texto de Entrada
inputSecret	Campo de Texto Cifrado de entrada
inputHidden	Campo de texto oculto
outputLabel	Despliega una etiqueta
outputLink	Enlace Html
outputFormat	Texto de salida con un formato establecido
outputText	Campo de texto o Mensaje de Salida
CommandButton	Botón
CommandLink	Enlace asociado a un botón
message	Muestra el mensaje más reciente para un componente
messages	Muestra todos los mensajes
graphicImage	Indica una Imagen
SelectOneListBox	Selección simple dentro de una lista
SelectOneMenu	Selección simple para un menú
SelectOneRadio	Selección dentro varios Radio Botones
selectBooleanCheckBox	CheckBox
selectManyCheckbox	Conjunto de CheckBox
selectManyListbox	Selección múltiple dentro de una lista
selectManyMenu	Selección múltiple dentro de un menú
dataTable	Tabla de datos
column	Columna de un dataTable

Tabla 1. 12. Etiquetas html la Tecnología JSF

Autor: Byron Delpino

1.14.2.1. Formulario

Genera un formulario a través de la etiqueta **h:form.** Esta contiene todo tipo de etiquetas: botones, menús, lista, campos de texto.

<h:form>

CODIGO(botones, campos de texto, listas)

</h:form>

1.14.2.2. Etiquetas de Entrada

Son etiquetas utilizadas para la entrada de texto desde un formulario.

- ✓ **h:inputText**
- ✓ **h:inputSecret**
- ✓ **h:inputTextarea**

Atributos

Atributo	Definición
style	Estilo de la etiqueta ejm: color, fondo(background), tamaño de letra(font-size)
maxlength	Número máximo de caracteres a ingresar en el campo de texto
Size	Tamaño de la etiqueta
rows	Número de Filas de area de texto (inputAreaText)
redisplay	Despliega en pantalla el texto ingresado

Tabla 1. 13. Atributos de etiquetas de Entrada

Autor: Byron Delpino

Ejemplos

Ejemplo	Código	Resultado
Campo de Texto	**<h:inputText value: "ESPE" /h>**	ESPE
Campo de Texto de tamaño 10	**<h:inputText value="REDES" size="10" />**	REDES
Campo de texto con letras azules	**<h:inputText value="ingrese…" style="color:BLUE"/>**	ingrese…
Campo de Texto de fondo amarillo y letras verdes de tamaño 100px	**<h:inputText value="Electrónica" style="color:GREEN;background:YELLOW;font-size:100px" />**	Electrónica
Campo de texto cifrado que despliega el carácter ingresado	**<h:inputSecret value="Electrónica" redisplay="true" />**	••••••••••
Área de Texto de 10 filas con fondo verde	**<h:inputTextarea value="Electrónica" rows="10" style="background:GREEN" />**	Electrónica
Campo de texto cuyo valor se almacena en el atributo nombre del Bean usuario.	**<h:inputText value="#{usuario.nombre}"/>**	
Campo de texto	**<h:inputSecret**	

cuyo valor se almacena en el atributo clave del Bean usuario.	**value="#{usuario.clave}"/>**

Tabla 1. 14. Utilización de etiquetas de Entrada

Autor: Byron Delpino

1.14.2.2. Etiquetas de Salida

Son etiquetas utilizadas para desplegar texto o imágenes en el formulario.

- ✓ **h:outputLabel**
- ✓ **h:outputLink**
- ✓ **h:outputText**
- ✓ **h:outputFormat**
- ✓ **h:graphicImage**
- ✓ **h:outputLink**

Atributos

Atributo	Definición
style	Estilo de la etiqueta ejm: color, fondo(background), tamaño de letra(font-size)
maxlength	Número máximo de caracteres a ingresar en el campo de texto

Tabla 1. 15. Atributos de etiquetas de Salida

Autor: Byron Delpino

Ejemplos

Ejemplo	Código	Resultado
Etiqueta que despliega mensaje “hola” con tamaño de 100px	**<h:outputLabel value="hola" style="font-size:100px"/>**	hola
Mensaje de salida cuyo texto es de color rojo y tamaño 20px	**<h:outputText value="Escuela Politécnica del Ejército" style="font-size:20px;color:RED"/>**	Escuela Politécnica del Ejército
Mensaje de salida, que despliega el valor del atributo nombre del Bean usuario.	**<h:outputText value="#{usuario.nombre}"/>**	Diana
Texto de salida con formato establecido, que toma los valores establecidos dentro de un parámetro añadido a dicho componente	**<h:outputFormat value="{0} es la capital de {1}, país de {2}">** **<f:param value="Quito"/>** **<f:param value="Ecuador"/>** **<f:param value="Sudamérica"/>** **</h:outputFormat>**	Quito es la capital de Ecuador, país de Sudamérica
Imagen desplegada con un ancho 1000, la misma fue obtenida desde un sitio web.	**<h:graphicImage value="http://www.espe.edu.ec/portal/files/sitiogestion2009/ESPE%20en%20cifras_archivos_/image001.jpg"**	

	width="1000"/>	
Mensaje de texto **www.google.com** que representa un enlace hacia **http://www.google.com.ec**.	**<h:outputLink value="http://www.google.com.ec">** **<h:outputText value="www.google.com.ec"/>** **</h:outputLink>**	www.google.com.ec

Tabla 1. 16. Utilización de etiquetas de Salida

Autor: Byron Delpino

1.14.2.3. Botones

Etiquetas que invocan una acción cuando se realiza determinado evento. Ejm: Si se presiona un botón se ejecutará una acción sobre el formulario.

- ✓ **h:commandButton**
- ✓ **h:CommandLink**

Atributos

Atributo	Definición
action	Define un método o acción a ser invocado
image	Imagen que va sobre el botón
value	Etiqueta o texto que va sobre el botón
type	hcommandButton { -submit, -reset, -button hCommandLink { -text/html, -image/gif
disabled	Deshabilita el botón

style	Estilo de la etiqueta ejm: color, fondo(background), tamaño de letra(font-size)

Tabla 1. 17. Atributos de Botones

Autor: Byron Delpino

Ejemplos

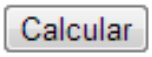
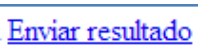
Ejemplo	Código	Resultado
Botón tipo submit	**<h:commandButton value="Enviar valores" type="submit"/>**	
Botón tipo reset con estilo de letra oblicua	**<h:commandButton value="Resetear" type="reset" style="font-style: oblique"/>**	
Botón con texto Calcular que al presionarlo ejecuta el método cal del Bean usuario	**<h:commandButton value="Calcular" action="#{usuario.cal}">**	
Enlace asociado a un botón	**<h:commandLink> <h:outputText value="Enviar resultado"/> </h:commandLink>**	

Tabla 1. 18. Utilización de Botones

Autor: Byron Delpino

1.14.2.4. Componentes de selección

Etiquetas que permiten la selección de uno o varios elementos.

- ✓ **h:selectBooleanCheckbox**
- ✓ **h:selectManyCheckBox**
- ✓ **h:selectOneRadio**

- ✓ **h:selectOneListBox**
- ✓ **h:selectManyListBox**
- ✓ **h:selectOneMenu**
- ✓ **h:selectManyMenu**

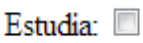
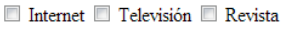
Atributos

Atributo	Definición
disabledClass	Deshabita el componente (selectOneRadio, selectManyCheckBox)
enabledClass	Habilita el componente (selectOneRadio, selectManyCheckBox)
layout	Especifica la orientación de los componentes lineDirection=Horizontal pageDirection=Vertical (selectOneRadio, selectManyCheckBox)

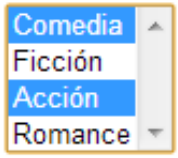
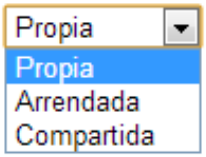
Tabla 1. 19. Atributos de Componentes de Selección

Autor: Byron Delpino

Ejemplos

Ejemplo	Código	Resultado
CheckBox Booleano, cuyo valor se almacena en el atributo estudio del Bean usuario.	**<h:selectBooleanCheckbox value="{usuario.estudio}"/>**	
Mensaje de texto “Estudia:” seguido de un CheckBox Booleano, cuyo valor se almacena en el atributo estudio del Bean usuario.	**<h:outputText value="Estudia: "/>** **<h:selectBooleanCheckbox value="{usuario.estudio}"/>**	
CheckBox múltiple cuyos	**<h:selectManyCheckbox**	

ítems seleccionados se almacenan en el atributo informa del Bean usuario.	**value="#{usuario.informa}"** **>** **<f:selectItem** **itemValue="internet"** **itemLabel="Internet"/>** **<f:selectItem** **itemValue="tv"** **itemLabel="Televisión"/>** **<f:selectItem** **itemValue="revista"** **itemLabel="Revista"/>** **</h:selectManyCheckbox>**	
Radio Botón que permite seleccionar el género, cuyo valor se almacena en el atributo genero del Bean usuario.	**<h:selectOneRadio** **value="#{usuario.genero}">** **<f:selectItem** **itemValue="hombre"** **itemLabel="Hombre"/>** **<f:selectItem** **itemValue="mujer"** **itemLabel="Mujer"/>** **</h:selectOneRadio>**	Hombre Mujer
Despliegue de lista de elementos, donde se seleccionará uno de ellos, dicho valor se almacenará en el atributo deporte del Bean usuario.	**<h:selectOneListbox** **value="#{usuario.deporte}">** **<f:selectItem** **itemValue="Fútbol"** **itemLabel="Fútbol"/>** **<f:selectItem** **itemValue="Basquet"** **itemLabel="Basquet"/>** **<f:selectItem** **itemValue="Volley"** **itemLabel="Volley"/>** **<f:selectItem**	

	itemValue="Karate" **itemLabel="Karate"/>** **</h:selectOneListbox>**	
Despliegue de lista de elementos, donde se seleccionará varios de ellos, dicho valor se almacenará en el atributo deporte del Bean usuario.	**<h:selectManyListbox value="#{usuario.peliculas}" >** **<f:selectItem itemValue="Comedia" itemLabel="Comedia"/>** **<f:selectItem itemValue="Ficción" itemLabel="Ficción"/>** **<f:selectItem itemValue="Acción" itemLabel="Acción"/>** **<f:selectItem itemValue="Romance" itemLabel="Romance"/>** **</h:selectManyListbox>**	
Menú de elementos, donde el valor del elemento seleccionado se almacena en el atributo vivienda del Bean usuario.	**<h:selectOneMenu value="#{usuario.vivienda}" >** **<f:selectItem itemValue="Propia" itemLabel="Propia"/>** **<f:selectItem itemValue="Arrendada" itemLabel="Arrendada"/>** **<f:selectItem itemValue="Compartida" itemLabel="Compartida"/>** **</h:selectOneMenu>**	

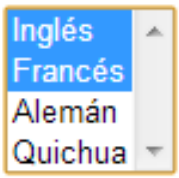
Menú de elementos, donde los valores de los elementos seleccionados se almacenan en el atributo idiomas del Bean usuario.	**<h:selectManyMenu value="#{usuario.idiomas}">** **<f:selectItem itemValue="Inglés" itemLabel="Inglés"/>** **<f:selectItem itemValue="Francés" itemLabel="Francés"/>** **<f:selectItem itemValue="Alemán" itemLabel="Alemán"/>** **<f:selectItem itemValue="Quichua" itemLabel="Quichua"/>** **</h:selectManyMenu>**	

Tabla 1. 20. Utilización de Componentes de Selección

Autor: Byron Delpino

1.15. Convertidores y Validadores

Son elementos JSF que permiten la conversión y validación de los valores recibidos y almacenados en los componentes JSF.

1.15.1. Conversión

El proceso de conversión se da cuando se transforma un determinado tipo de dato en otro. La conversión se presenta en:

- ✓ **Obtención entradas de usuario:** Conversión del valor de un componente (inputText, inputSecret, inputTextarea) de tipo String a número (entero, doble) o fecha.

  String => Numero/Fecha.

- ✓ **Generación de la Presentación:** Cuando un dato almacenado de tipo numérico o fecha se presenta en un componente de salida (outputText) que es de tipo String. Numero/Fecha =>String

Para la conversión de datos se utiliza las siguientes etiquetas Core:

- ✓ **f:convertNumber**
- ✓ **f:convertDateTime**

Atributos

f:convertNumber

Atributo	Tipo	Definición
type	String	Conversión de número a String con formato number(por defecto), currency (€), porcent (%).
maxFractionDigits	int	Número máximo de dígitos en la parte fraccional al convertir String a número.
minFractionDigits	int	Número mínimo de dígitos en la parte fraccional al convertir String a número.
maxIntegerDigits	int	Número máximo de dígitos en la parte entera al convertir String a número.
minIntegerDigits	int	Número mínimo de dígitos en la parte entera al convertir String a número.
pattern	String	Formato de patrón en la conversión de String a Número

Tabla 1. 21. Atributos de de la convertidor numérico

Autor: Byron Delpino

f: convertDateTime

Atributo	Tipo	Definición
type	String	Conversión de número a Fecha con date y/o time.
pattern	String	Formato de patrón en la conversión de String a Fecha.

Tabla 1. 22. Atributos del convertidor tipo fecha

Autor: Byron Delpino

Ejemplos

Ejemplo	Código
Conversión a Fecha en el formato MM/yyyy	<f:convertDateTime pattern="MM/yyyy"/>
Conversión a Número con un máximo de 2 dígitos en la parte fraccional	<f:convertNumber maxFractionDigits="2" />
Conversión a número formato moneda.	<f:convertNumber type="currency" />

Tabla 1. 23. Utilización de Convertidores

Autor: Byron Delpino

1.15.2. Validación

La validación de datos se da en los valores ingresados por parte del usuario en un componente de entrada. Se puede controlar parámetros como el número mínimo o máximo de caracteres ingresados, o el rango de un valor.

Para la validación de datos se utiliza las siguientes etiquetas Core:

- ✓ **f:validateLength**
- ✓ **f:validateDoubleRange**
- ✓ **f:validateLongRange**

Atributos

Atributo	Definición
minimum	Valor mínimo
maximum	Valor Máximo

Tabla 1. 24. Atributos de validadores

Autor: Byron Delpino

Ejemplos

Ejemplo	Código
Validación que un dato tipo doble esté entre 0 y 20	<f:validateDoubleRange minimum="0" maximum="20"/>
Validación para que la longitud de un dato este entre 1 y 10 dígitos	<f:validateLength minimum="1" maximum="10"/>

Tabla 1. 25. Utilización de Validadores

Autor: Byron Delpino

1.16. PrimeFaces

PrimeFaces es una librería de componentes visuales basados en código abierto creado por la compañía turca Prime Technology.

Esta tecnología es un buen complemento para JSF en cuanto a la presentación, ya que permite el desarrollo de aplicaciones web con una interfaz gráfica mucho más llamativa para el usuario. Primefaces incorpora 117 componentes:

- **Componentes Básicos:** Botones, campos de texto, componentes de selección, imágenes, tablas.
- **Componentes Extras:** Calendarios, Editores de texto, Cuadros Estadísticos, Exportación de Datos a Word, Excel, incorporación con googlemaps.

1.16.1. Descarga e Instalación

- ✓ En la página de PrimeFaces http://primefaces.org/downloads.html, se puede descargar la librería.
- ✓ Para instalar PrimeFaces, una vez que se ha creado la aplicación JSF, se añade la librería PrimeFaces.jar.

Figura 1. 57. Incorporación de la librería PrimeFaces.jar

Autor: Byron Delpino

- ✓ En la interfaz gráfica (.xhtml) se añade la etiquetas Primefaces: **xmlns:p="http://primefaces.org/ui"**

1.16.2. Ejemplos Básicos

Ejemplo	Código	Resultado
Campo de texto cuyo valor se almacena en el atributo nota del Bean Alumno.	**<p:inputText value="#{Alumno.nota}">**	
Despliegue de una imagen	**<p:graphicImage value="ESPE.gif" width="80" />**	
Botón con texto Ingresar que al presionarlo ejecuta el método ingresar del Bean Alumno	**<p:commandButton action="#{Alumno.ingresar()}" value="Ingresar" />**	Ingresar
Menú de elementos, donde el valor del elemento seleccionado se almacena en el atributo facultad del Bean Alumno	**<p:selectOneMenu value="#{Alumno.facultad}"> <f:selectItem itemValue="Redes" itemLabel="Electrónica en Redes"/> <f:selectItem itemValue="Telecomunicaciones" itemLabel="Electrónica en Telecomunicaciones"/>**	Electrónica en Telecomunicaciones Electrónica en Redes Electrónica en Telecomunicaciones Electrónica en Automatización y Control

	<f:selectItem itemValue="Control" itemLabel="Electrónica en Automatización y Control"/> </p:selectOneMenu>	
Despliegue de un calendario, el valor obtenido del mismo sigue el formato días, mes,año.	**<p:calendar value="#{Alumno.fecha}" pattern="dd/MM/yyyy"/>**	

Tabla 1. 26. Ejemplos Básicos del uso de componentes PrimeFaces

Autor: Byron Delpino

1.17. Laboratorios

1.17.1. Navegación Estática y Dinámica

Este laboratorio se centra en los conceptos de navegación estática y dinámica y el empleo de Managed Beans.

1.17.1.1. Diagrama de Bloques

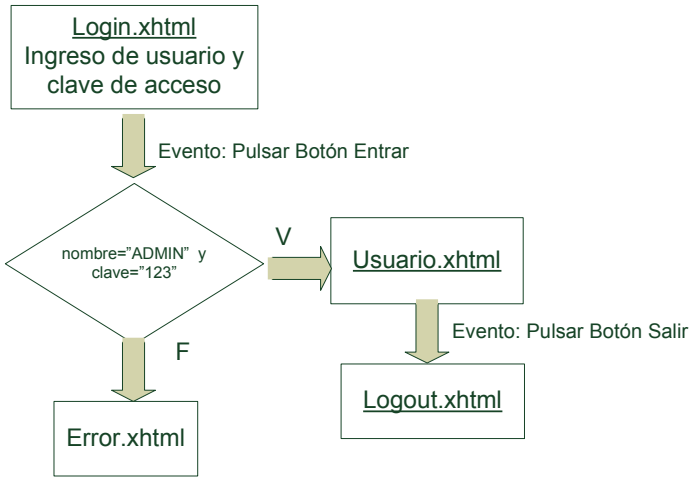


Figura 1. 58. Diagrama de Bloques, Navegación Estática y Dinámica

Autor: Byron Delpino

1.17.1.2. Programación Java

- En Netbeans, crear una aplicación java, New File – Java Web – Web Aplication, Nombre y Ubicación de la aplicación- selección de Servidor GlassFish y JEE 6 Web –Framework: JavaServer Faces –Finish.
- Se crea los archivos .xhtml (Login, Usuario, Error, Logout), .java (IngresoBean) y .xml(face-config).

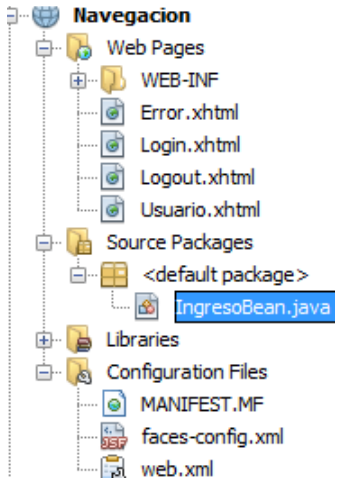


Figura 1. 59. Creación de ficheros xhtml, java y xml

Autor: Byron Delpino

1.17.1.2.1. Definición de Variables

Nombre	Tipo	Definición
nombre	String	Variable que representa el atributo nombre del Bean usuario
clave	String	Variable que representa el atributo clave del Bean usuario
Nombre	String	Variable auxiliar utilizada para establecer el valor del atributo nombre
Clave	String	Variable auxiliar utilizada para establecer el valor del atributo clave

Tabla 1. 27. Definición de Variables utilizadas en el Bean usuario

Autor: Byron Delpino

1.17.2. Reconocimiento de los componentes en un proyecto JSF

Este laboratorio se centra en el uso de las etiquetas jsf, el uso de Managed Beans y navegación estática.

La aplicación web corresponde a una encuesta en donde el usuario deberá ingresar su nombre, estado civil, deportes, actividades favoritas, idiomas; una vez que este insertó todos sus datos en un formulario, en otra página se desplegará dicha información.

1.17.2.1. Diagrama de Bloques

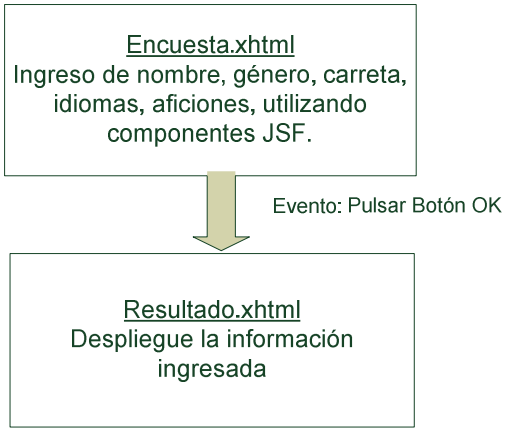


Figura 1. 60. Diagrama de Bloques, Reconocimiento componentes JSF

Autor: Byron Delpino

1.17.2.2. Programación Java

- En Eclipse, New File – Dynamic Web Project, Nombre y Ubicación de la aplicación- selección de Servidor JBoss 7.1 Runtime –Configuración: JavaServer Faces v 2.1 Project –Next –Next – Activar Generate web.xml deployment descriptor– Finish.

- Se crea los archivos .xhtml (Encuesta, Resultado), .java (UsuarioBean) y .xml(face-config).

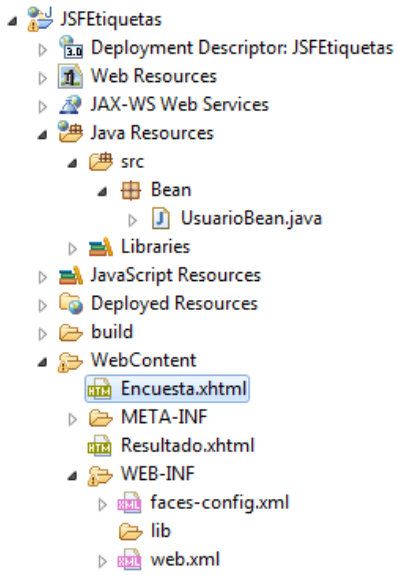


Figura 1. 61. Creación de ficheros xhtml, java y xml

Autor: Byron Delpino

1.17.2.2.1. Definición de Variables

Nombre	Tipo	Definición
nombre	String	Variable que representa el atributo nombre del Bean usuario
genero	String	Variable que representa el atributo genero del Bean usuario
facultad	String	Variable que representa el atributo facultad del Bean usuario
idiomas	Arreglo de Strings	Variable que representa el atributo idiomas del Bean usuario
peliculas	Arreglo de Strings	Variable que representa el atributo películas del Bean usuario
informacion	Arreglo de Strings	Variable que representa el atributo informacion del Bean usuario
comentarios	Strings	Variable que representa el atributo comentarios del Bean

		usuario
Nombre	String	Variable auxiliar utilizada para establecer el valor del atributo nombre
Genero	String	Variable auxiliar utilizada para establecer el valor del atributo genero
Facultad	String	Variable auxiliar utilizada para establecer el valor del atributo facultad
Idiomas	Arreglo de Strings	Variable auxiliar utilizada para establecer el valor del atributo idiomas
Peliculas	Arreglo de Strings	Variable auxiliar utilizada para establecer el valor del atributo películas
Informacion	Arreglo de Strings	Variable auxiliar utilizada para establecer el valor del atributo información
Comentarios	String	Variable auxiliar utilizada para establecer el valor del atributo comentarios
cadena	String	Variable auxiliar utilizada para realizar la concatenación de variables tipo arreglo de Strings
arreglo	Arreglo de Strings	Variable auxiliar tipo arreglo de Strings utilizada para realizar la concatenación
ListaIdiomas	String	Variable auxiliar que representa el resultado de la concatenación del Arreglo de Strings idiomas
ListaPeliculas	String	Variable auxiliar que representa el resultado de la concatenación

		del Arreglo de Strings peliculas
ListaInformacion	String	Variable auxiliar que representa el resultado de la concatenación del Arreglo de Strings informacion

Tabla 1. 28. Definición de Variables utilizadas en el Bean usuario

Autor: Byron Delpino

1.17.3. Convertidores y validadores

Utilización de convertidores y validadores, para el ingreso de datos del estudiante (nombre, cédula, notas, pago de derechos, fecha de evaluación).

1.17.3.1. Diagrama de Bloques

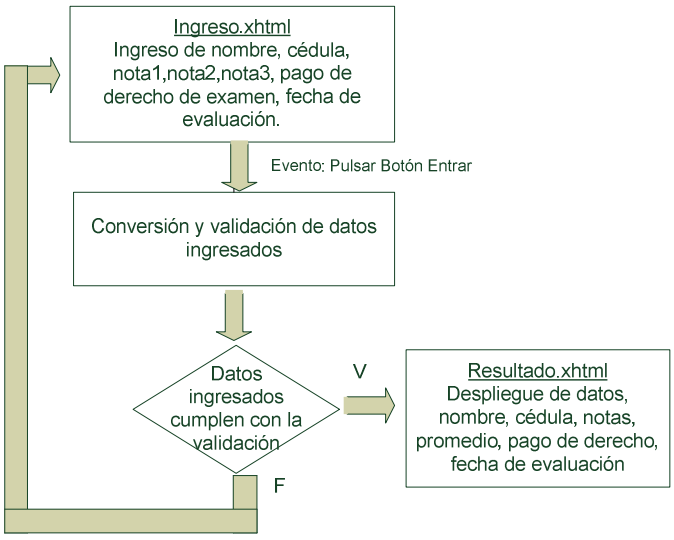


Figura 1. 62. Diagrama de Bloques, Convertidores y Validadores

Autor: Byron Delpino

1.17.3.2. Programación Java

- En Eclipse, New File – Dynamic Web Project, Nombre y Ubicación de la aplicación- selección de Servidor JBoss 7.1 Runtime –Configuración: JavaServer Faces v 2.1 Project –Next –Next – Activar Generate web.xml deployment descriptor– Finish.

- Se crea los archivos .xhtml (Ingreso, Resultado), .java (Estudiante) y .xml(face-config).

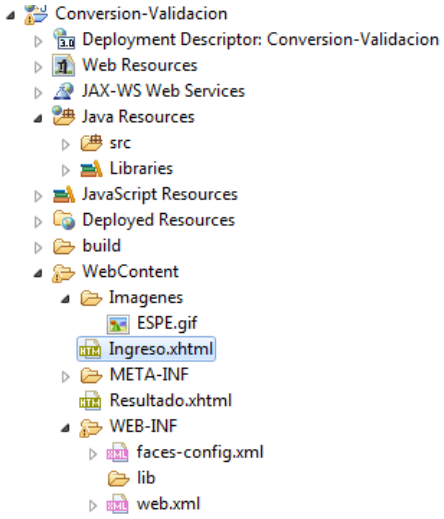


Figura 1. 63. Creación de ficheros xhtml, java y xml

Autor: Byron Delpino

1.17.3.2.1. Definición de Variables

Nombre	Tipo	Definición
nombre	String	Variable que representa el atributo nombre del Bean alumno
cedula	String	Variable que representa el atributo cedula del Bean alumno
nota1	Doble	Variable que representa el atributo nota2 del Bean alumno
nota2	Doble	Variable que representa el atributo

		nota3 del Bean alumno
nota3	Doble	Variable que representa el atributo nota1 del Bean alumno
pagos	Doble	Variable que representa el atributo pagos del Bean alumno
fecha	Date	Variable que representa el atributo fecha del Bean alumno
promedio	Doble	Variable utilizada para obtener el promedio del alumno
Nombre	String	Variable auxiliar utilizada para establecer el valor del atributo nombre
Cedula	String	Variable auxiliar utilizada para establecer el valor del atributo cedula
Nota1	Doble	Variable auxiliar utilizada para establecer el valor del atributo nota1
Nota2	Doble	Variable auxiliar utilizada para establecer el valor del atributo nota2
Nota3	Doble	Variable auxiliar utilizada para establecer el valor del atributo nota3
Pagos	Doble	Variable auxiliar utilizada para establecer el valor del atributo pagos
Fecha	Date	Variable auxiliar utilizada para establecer el valor del atributo fecha

Tabla 1. 29. Definición de Variables del Bean alumno

Autor: Byron Delpino

1.17.4. Uso de Listas y PrimeFaces

Laboratorio centrado en el empleo de Listas, validación, conversión y componentes PrimeFaces. El usuario ingresa información de un listado de alumnos (nombre, facultad, fecha de evaluación y nota), los datos ingresados se validan y se despliegan en una tabla.

1.17.4.1. Diagrama de Bloques

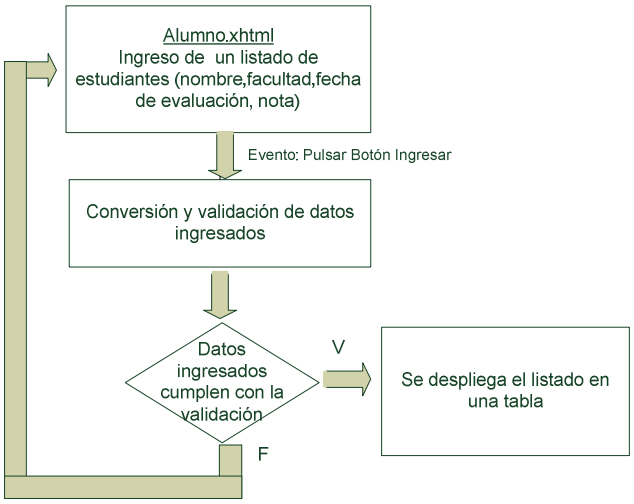


Figura 1. 64. Diagrama de Bloques, Uso de Listas, PrimeFaces

Autor: Byron Delpino

1.17.4.2. Programación Java

- En Netbeans, crear una aplicación java, New File – Java Web – Web Aplication, Nombre y Ubicación de la aplicación- selección de Servidor GlassFish y JEE 6 Web –Framework: JavaServer Faces –Finish.

- Se crea el archivo .xhtml (Alumno), .java (Alumno, ControlAlumnos) y .xml(face-config). Además se añade la librería PrimeFaces.jar.

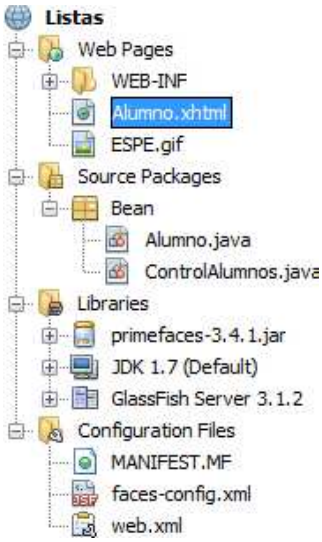


Figura 1. 65. Creación de ficheros xhtml, java y xml

Autor: Byron Delpino

Se utiliza dos Beans representados en las clases Alumno.java y ControlAlumnos.java, el primero se utiliza para el ingreso de datos como nombre, fecha de evaluación, facultad, y nota, el segundo se emplea para la incorporación de varios objetos tipo alumno, de esta forma se puede crear un listado de alumnos.

1.17.4.2.1. Definición de Variables

Alumno.java

Nombre	Tipo	Definición
nombre	String	Variable que representa el atributo nombre del Bean alumno
facultad	String	Variable que representa el atributo facultad del Bean alumno
nota	Doble	Variable que representa el atributo nota del Bean alumno
fecha	Date	Variable que representa el atributo fecha del Bean alumno
Nombre	String	Variable auxiliar utilizada para establecer el valor del atributo nombre

Facultad	String	Variable auxiliar utilizada para establecer el valor del atributo facultad
Nota	Doble	Variable auxiliar utilizada para establecer el valor del atributo nota
Fecha	Date	Variable auxiliar utilizada para establecer el valor del atributo fecha

Tabla 1. 30. Definición de Variables del Bean alumno

Autor: Byron Delpino

ControlAlumnos.java

Nombre	Tipo	Definición
alum	Alumno	Objeto tipo Alumno que permite el ingreso de datos de un estudiante
alumnos	Listado de Alumno	Listado de Objetos tipo Alumno que permite el ingreso de varios estudiantes.
alumnox	Alumno	Objeto auxiliar utilizado para establecer el valor del atributo alum
alumnosx	Listado de Alumno	Listado de Objetos auxiliar utilizado para establecer el valor del atributo alumnos

Tabla 1. 31. Definición de Variables del Bean controlAlumno

Autor: Byron Delpino

1.17.5. Lectura y Escritura del puerto USB desde JSF

Este laboratorio se centra en la escritura y lectura del puerto usb utilizando un dipswitch y un banco de leds.

Tanto los materiales, como la programación del microcontrolador se encuentran en el laboratorio **1.6.2 "Lectura y Escritura del puerto USB utilizando leds y dipwitch",** por lo que para el desarrollo del presente laboratorio únicamente se hará hincapié en la programación JSF.

1.17.5.1. Diagrama de Bloques

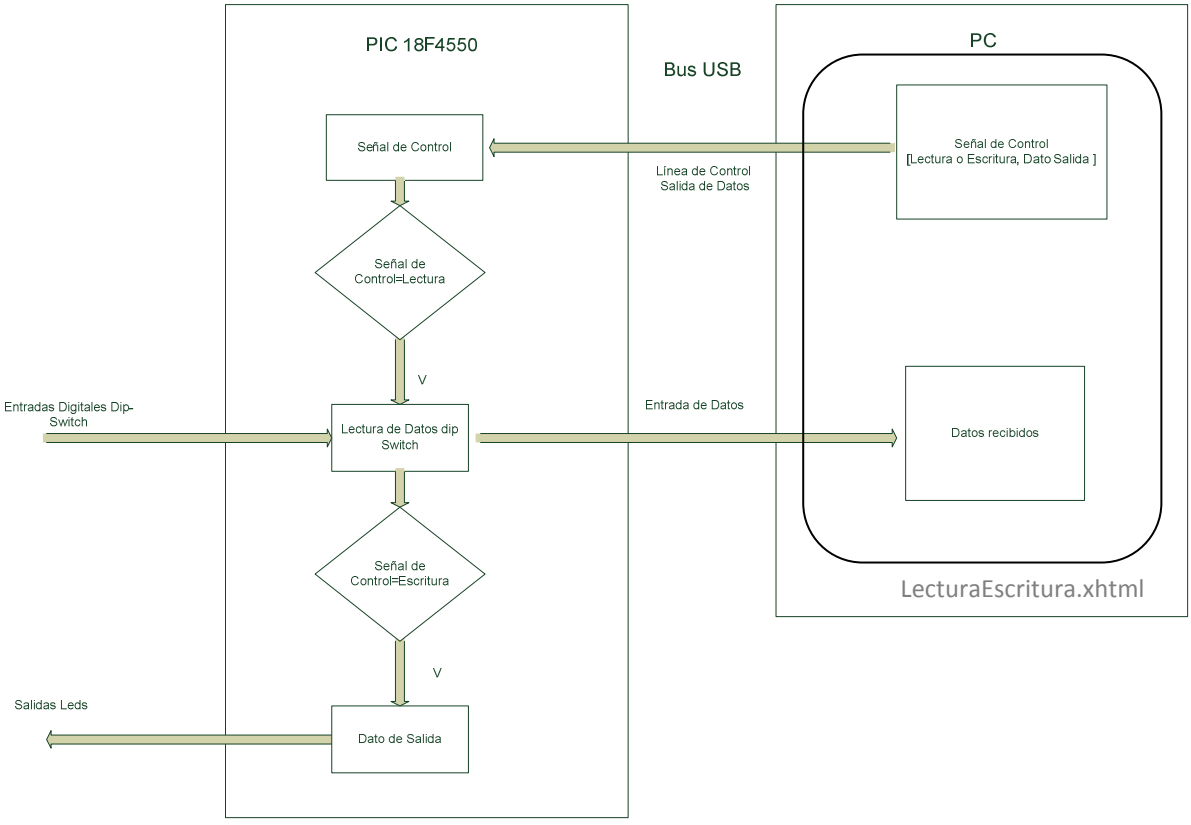


Figura 1. 66. Diagrama de Bloques, lectura y escritura de datos USB

Autor: Byron Delpino

1.17.5.2. Programación Java

- En Eclipse, New File – Dynamic Web Project, Nombre y Ubicación de la aplicación- selección de Servidor JBoss 7.1 Runtime –Configuración: JavaServer Faces v 2.1 Project –Next –Next – Activar Generate web.xml deployment descriptor– Finish.

- Se crea los archivos .xhtml (LecturaEscritura) .java (UsbBean) y .xml(face-config). Además se añade la librería PrimeFaces.jar y jpicusb.jar.

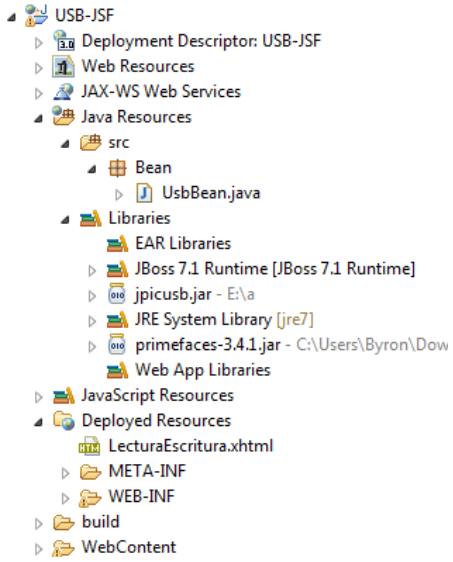


Figura 1. 67. Creación de ficheros xhtml, java y xml

Autor: Byron Delpino

1.17.5.2.1. Definición de Variables

Nombre	Tipo	Definición
recepcion	String	Variable que representa el atributo recepcion del Bean usb
envio	Arreglo de Strings	Variable que representa el atributo envio del Bean

		usb
Recepcion	String	Variable auxiliar empleada para establecer el valor del atributo recepción
Envio	Arreglo de Strings	Variable auxiliar empleada para establecer el valor del atributo Envio
lectura	Byte	Comando para Lectura de datos USB
escritura	Byte	Comando para Escritura de datos USB
out	Arreglo de Bytes	Envía la variable lectura o escritura a través del Puerto USB, además envía la variable DATO
VALOR	Byte	Dato a ser enviado por el puerto USB
dato	String	Dato recibido desde puerto USB
a	Entero	Variable auxiliar para obtener el dato a ser escrito en el puerto USB

Tabla 1. 32. Definición de Variables del Bean usb

Autor: Byron Delpino

1.17.6. Lectura y Escritura del puerto USB desde JSF-PrimeFaces

Lectura y Escritura del puerto USB utilizando sensores de temperatura y ventiladores.

Tanto los materiales, como la programación del microcontrolador se encuentran en el laboratorio **1.6.3 “Lectura y Escritura del puerto USB utilizando sensores y**

ventiladores", por lo que para el desarrollo del presente laboratorio únicamente se hará hincapié en la programación JSF.

1.17.6.1. Diagrama de Bloques

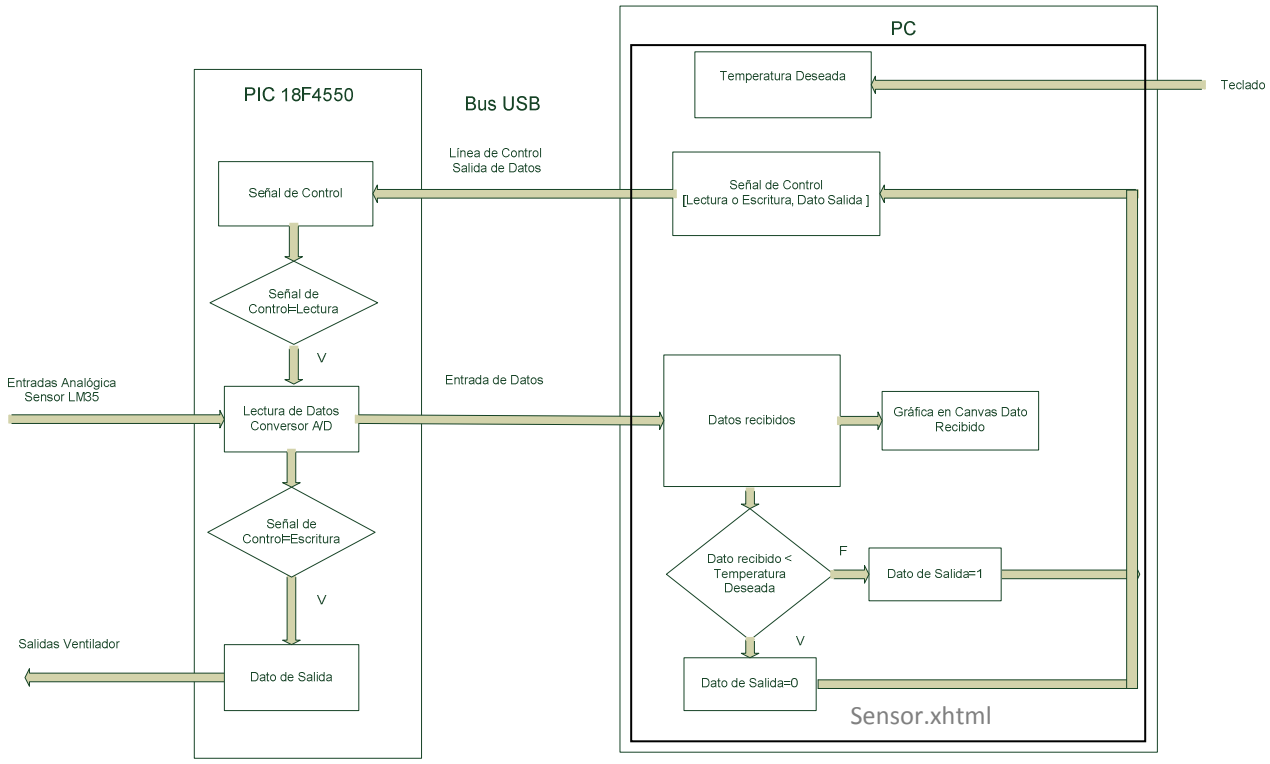


Figura 1. 68. Diagrama de Bloques, lectura y escritura de datos USB

Autor: Byron Delpino

1.17.6.2. Programación Java

- En Netbeans, crear una aplicación java, New File – Java Web – Web Aplication, Nombre y Ubicación de la aplicación- selección de Servidor GlassFish y JEE 6 Web –Framework: JavaServer Faces –Finish.

- Se crea el archivo .xhtml (Sensor), .java (USBLm35) y .xml(face-config). Además se añade la librería PrimeFaces.jar y jpicusb.jar.

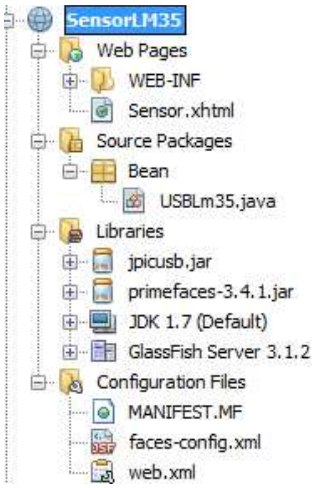


Figura 1. 69. Creación de ficheros xhtml, java y xml

Autor: Byron Delpino

1.17.6.2.1. Definición de Variables

Nombre	Tipo	Definición
recepcion	String	Variable que representa el atributo recepcion del Bean usb
seteo	String	Variable que representa el atributo seteo del Bean usb
ventilador	String	Variable que representa el atributo ventilador del Bean usb
obtener	MeterGaugeChartModel	Variable utilizada para el manejo del componente primefaces medidor de temperatura
intervals	List<Number>	Variable usada para incorporar los intervalos de temperatura en el componente meterGaugeChart
temperatura	Doble	Variable auxiliar utilizada para obtener el valor de la temperatura actual del sensor
Nota	Doble	Variable auxiliar utilizada para establecer el valor del atributo nota
Recepcion	String	Variable auxiliar empleada para

		establecer el valor del atributo recepción
Seteo	String	Variable auxiliar utilizada para establecer el valor del atributo seteo
Ventilador	String	Variable auxiliar empleada para establecer el valor del atributo ventilador
VIDyPID	String	VID y PID del PIC
instancia	Entero	Instancia
lectura	Byte	Comando para Lectura de datos USB
escritura	Byte	Comando para Escritura de datos USB
out	Arreglo de Bytes	Envía la variable lectura o escritura a través del Puerto USB, además el valor a enviar por puerto USB
dato	String	Variable auxiliar para recibir dato leído desde el puerto usb
valseteo	String	Variable auxiliar empleada para obtener el valor de la temperatura deseada

Tabla 1. 33. Definición de Variables del Bean usb

Autor: Byron Delpino

JAVA 2 MICRO EDITION (J2ME)

1.18. Introducción

A mediados de los años 90, el lenguaje de programación Java fue diseñado para el desarrollo de aplicaciones que controlaban electrodomésticos como lavadoras y refrigeradores, esto debido a la robustez e independencia de la plataforma donde estas se ejecutaban.

Con el desarrollo tecnológico en el campo de la informática y comunicaciones, Java se convirtió en el lenguaje por excelencia para implementar aplicaciones en varios ámbitos como servicios HTTP, manejo de base de datos, para ello Sun Microsystems lanzó las plataformas J2SE(Java 2 Standard Edition), para aplicaciones independientes y J2EE(Java 2 Enterprise Edition), centrada al entorno empresarial.

En la última década, con el gran avance de la telefonía celular, hoy en día es posible apreciar aplicaciones en dispositivos móviles que antes exclusivamente se podían ejecutar en un computador. En 1999 Sun desarrolló la plataforma J2ME, enfocada al entorno de dispositivos pequeños con capacidades limitadas en cuanto a memoria, procesamiento y pantalla gráfica.

Figura 1. 70. Logotipo plataforma Java Micro Edition

Fuente: www.erickcion.wordpress.com

Java Micro Edition, es una versión reducida de J2SE creada para adaptarse a las características de dispositivos que cuentan con reducidas capacidades gráficas y computacionales como teléfonos móviles, TDT (Televisión Digital Terrestre), reproductores Blu-ray, equipos multimedia digitales, PDAs, impresoras.

J2ME posee una mínima parte de las API (Interfaz de Programación de Aplicaciones) de Java, en total cuenta con 37 clases de la plataforma J2SE, proveniente de los paquetes java.lang, java.io, java.util, además utiliza las máquinas virtuales KVM (Kilo Virtual Machine) y CVM (Compact Virtual Machine), dependiendo de la capacidad del aparato, en lugar de JVM (Java Virtual Machine).

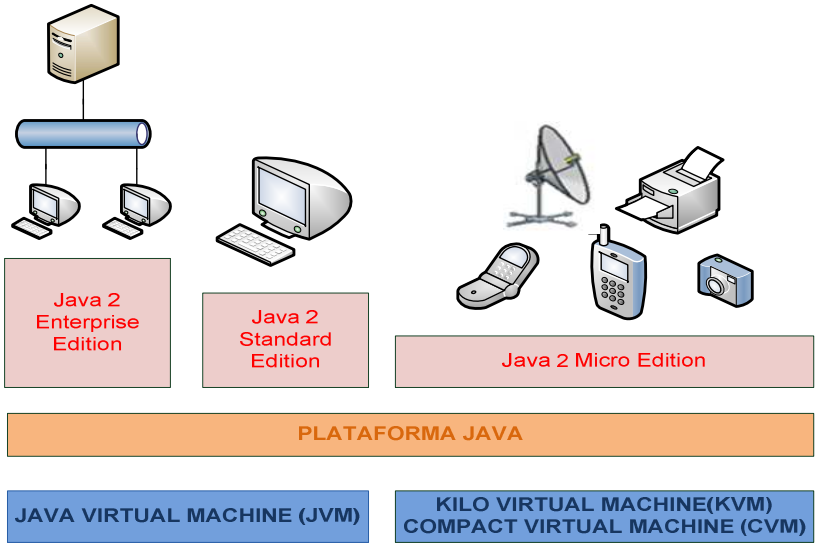


Figura 1. 71. Tecnologías de la plataforma JAVA

Autor: Byron Delpino

Los programas J2ME se denominan MIDLETS que van comprimidos en paquetes .jar, que contienen todos los ficheros que conforman la aplicación (nombre, versión, fabricante, imágenes, e/o).

Además del archivo .jar, una aplicación J2ME incluye el .jad, que almacena información sobre nombre de la aplicación, nombre y versión del MIDLET, tipo de perfil y configuración, nombre del .jar.

1.19. Arquitectura J2ME

La plataforma J2ME es un conjunto de tecnologías y librerías que soportan una gran variedad de mecanismos. J2ME está conformada por:

- Máquina Virtual
- Configuración
- Perfiles
- Paquetes Opcionales

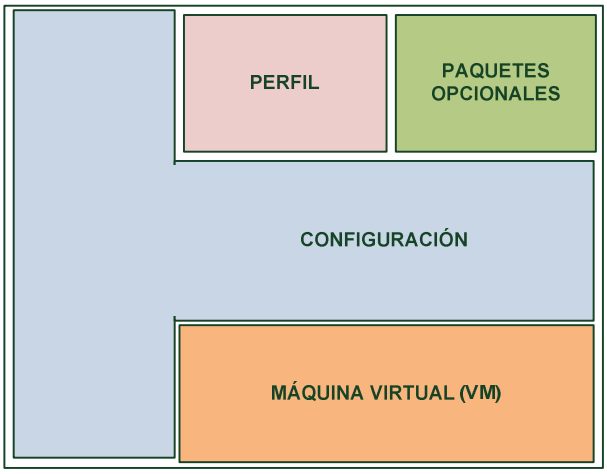


Figura 1. 72. Arquitectura Java 2 Micro Edition J2ME

Autor: Byron Delpino

1.19.1. Máquinas Virtuales

La Máquina virtual es un programa que proporciona independencia entre la plataforma JAVA con respecto al hardware y el sistema operativo S.O.

Entre sus tareas están:

- ✓ Interpretar el código intermedio (bytecode- código binario) de los programas java a código máquina.
- ✓ Comunicarse con el sistema operativo del dispositivo.
- ✓ Observar las reglas de seguridad y corrección del código del lenguaje java.

J2ME define dos tipos de máquinas virtuales, dependiendo de las capacidades del equipo:

- ✓ **KVM (Kilo Virtual Machine):** Máquina virtual para artefactos con grandes limitaciones computacionales (Celulares, PDA). Ocupa entre 40-80KB, no soporta coma flotante (decimales) ni JNI.

- ✓ **CVM (Compact Virtual Machine):** Utilizada para mecanismos que cuentan con mejores capacidades computacionales (sistemas satelitales, algunos electrodomésticos, sistemas de navegación de automóviles). Orientado para dispositivos con procesadores de 32 bits y más 2MB de memoria RAM. Soporta coma flotante, JNI, hilos.

1.19.2. Configuraciones

Contiene un mínimo de clases y bibliotecas que ofrecen la funcionalidad para un rango particular de dispositivos con características comunes.

Las configuraciones se describen según las capacidades de memoria del dispositivo:

- ✓ **CLDC (Connected Limited Device Configuration):** Configuración enfocada en instrumentos con memoria limitada y bajo procesamiento (ejemplos: teléfonos móviles, PDAs). CLDC define la máquina virtual KVM, los dispositivos que usan CLDC deben cumplir con los siguientes requisitos:

  - o Disponer entre 160KB-512KB de memoria total disponible.
  - o Procesador de 16 a 32 bits.
  - o Ofrecer bajo consumo de energía.
  - o Tener algún mecanismo para conectarse a la red.

  CLDC incluye paquetes como:

  - o **java.io**: Clases e Interfaces de E/S.
  - o **java.lang**: Clases básicas del lenguaje.
  - o **javax.microedition.io:** Clases e interfaces para conexión genérica CLDC.

- ✓ **CDC (Connected Device Configuration):** Configuración orientada para equipos con capacidades de memoria y procesamiento (ejemplos: decodificadores de televisión digital, televisores con internet, sistemas de navegación de automóviles). CDC define la máquina virtual CVM y está enfocada en instrumentos que poseen las siguientes características:

  - Procesador 32 bits.
  - Memoria RAM y ROM mayor a 2MB.
  - Conectividad a algún tipo de red.

  CDC incluye paquetes como:

  - **java.io**: Clases e Interfaces de E/S.
  - **java.lang**: Clases básicas del lenguaje.
  - **java.math**: Paquetes de matemáticas.
  - **java.net**: Clases e interfaces de red.
  - **java.security**: Clases e interfaces de seguridad.
  - **javax.microedition.io:** Clases e interfaces para conexión genérica CDC.

1.19.3. Perfiles

Los perfiles controlan el ciclo de vida de la aplicación y la interfaz del usuario, siendo un conjunto de APIs orientados a un ámbito de aplicación determinado. Un perfil identifica a un grupo de dispositivos por la función que cumplen (electrodomésticos, teléfonos, PDAs) y el tipo de aplicaciones que se ejecutan en ellos.

1.19.3.1. Perfiles CDC

- ✓ **Foundation Profile**

  Perfil que define una serie de APIs orientadas a dispositivos que carecen de una interfaz gráfica (ejemplo: decodificadores de televisión digital). Foundation Profile incluye la mayoría de APIs de J2SE salvo los paquetes java.awt y java.swing.

Foundation Profile incluye paquetes como:

- **java.lang**: Soporte del lenguaje java.
- **java.net**: Incluye soportes TCP/IP y conexiones HTTP.
- **java.io**: Clases reader y writer de J2SE.
- **java.text**: Incluye soportes para la internacionalización.
- **java.security**: Incluye códigos y certificados.

✓ **Personal Profile**

Perfil que incluye API orientada para aquellos dispositivos que poseen una interfaz gráfica y capacidades web.

Personal Profile incluye paquetes como:

- **java.awt**: Clases para crear GUIs con awt.
- **java.awt.datatransfer**: Clases e interfaces para transmitir datos entre aplicaciones.
- **java.awt.image:** Clases para crear y modificar imágenes.
- **javax.microedition.xlet**: Interfaces para la comunicación.

✓ **RMI Profile**

Este perfil es un subconjunto de las APIs de J2SE, con diversas limitaciones debido a las capacidades computacionales de los dispositivos móviles que implementan la CVM.

1.19.3.2. Perfiles CLDC

✓ **PDA Profile**

Pretende abarcar PDAs, tipo Palm, que poseen una pantalla, un puntero y una resolución mayor a 20000 pixeles.

- ✓ **Mobile Information Device Profile (MIDP)**

  Es el primer perfil definido por J2ME para equipos móviles (teléfonos y PDAs), MIDP está orientado a dispositivos con las siguientes características:

  - Reducida capacidad de procesamiento y memoria.
  - Conectividad limitada.
  - Baja capacidad gráfica

  MIDP incluye paquetes como:

  - **javax.microedition.lcdui**: Clases e Interfaces para GUIs.
  - **javax.microedition.rms:** Soporte para almacenamiento de datos.
  - **javax.microedition.midlet:** Clases de definición de la aplicación.
  - **javax.microedition.io:** Clases e interfaces de conexión genérica.
  - **java.io**: Clases e Interfaces de E/S.
  - **java.lang**: Soporte del lenguaje java.

  Las aplicaciones realizadas que utilizan MIDP se denominan MIDlets. MIDP facilita el desarrollo de aplicaciones intuitivas y gráficas optimizadas para pequeñas pantallas, permitiendo al usuario ingresar y observar datos que pueden ser almacenados, o conectarse a la red.

1.19.4. Paquetes Opcionales

Constituyen un conjunto de librerías que proporcionan una funcionalidad adicional a la aplicación J2ME, ejemplo:

- API para bluetooth, manejo de conectividad a través de bluetooth.
- Wireless Messaging API, envio y recepción de mensajes SMS.
- Mobile Media API, manejo de datos multimedia.

1.20. MIDlets

Un MIDlet es una aplicación J2ME, que es ejecutada bajo el perfil MIDP en equipos con poca capacidad gráfica, de procesamiento y de memoria.

La gestión de un MIDlet dentro de un dispositivo se da a través de un AMS (Application Management System), un software que reside dentro del equipo, encargado de ejecutar, pausar o destruir la aplicación.

1.20.1. Ciclo de Vida de un MIDlet

Una vez que el MIDlet ha sido descargado e instalado dentro del equipo, el AMS gestiona su ciclo de vida durante la ejecución. El ciclo de vida está conformado por tres estados:

- ✓ **Activo:** El MIDlet se está ejecutando.
- ✓ **Pausado:** El MIDlet no está actualmente en ejecución, pero puede pasar a ejecución al cambiar su estado a Activo.
- ✓ **Destruido:** El MIDlet no se está ejecutando ni puede transitar a otro estado, además se liberan todos los recursos ocupados por el MIDlet.

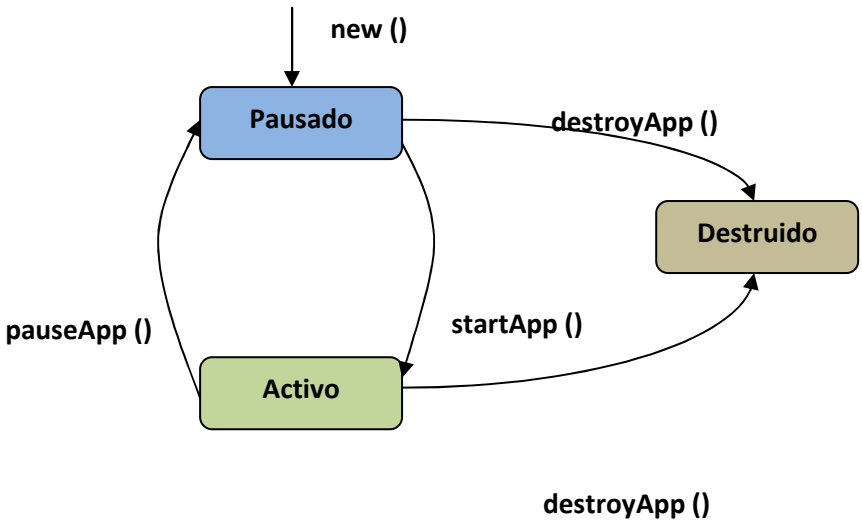


Figura 1. 73. Ciclo de Vida de un MIDlet

Autor: Byron Delpino

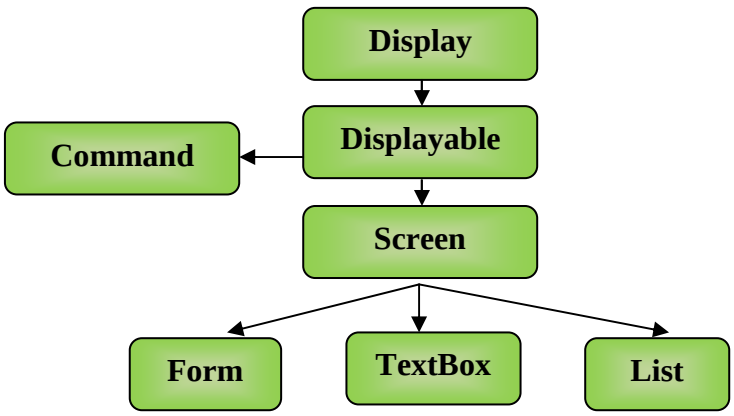
Un MIDlet puede cambiar de estado mediante una llamada a los métodos MIDlet.startApp(), MIDlet.pauseApp() o MIDlet.destroyApp().

Cuando el AMS gestiona el MIDlet, primero invoca al constructor del MIDlet (inicialización de variables), pasando al estado Pausado durante un corto periodo de tiempo, una vez que el dispositivo está preparado para ejecutar el MIDlet, el AMS llama al método MIDlet.startApp(), para entrar en el estado Activo. Durante este estado se puede transitar al estado Pausado a través del método MIDlet.pauseApp() por acción del usuario o para reducir el consumo de recursos del dispositivo. En los estados Activo o Pausado se puede ir al estado Destruido realizando una llamada al método MIDlet.destroyApp(), esto se da para liberar todos los recursos ocupados cuando haya finalizado la ejecución de la aplicación.

1.21. Interfaces gráficas de usuario

El perfil MIDP es el encargado de establecer parámetros de un equipo móvil como interfaces de usuario, sonidos, almacenamiento.

El paquete javax.microedition.lcdui, proporciona los mecanismos para desarrollar interfaces de usuario. A continuación se detalla las clases del paquete javax.microedition.lcdui que van a ser objeto de estudio.



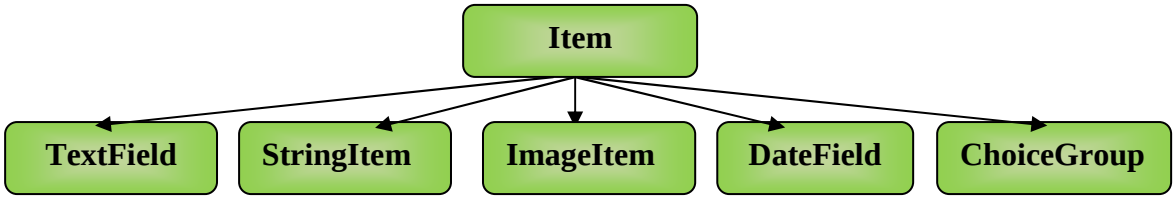


Figura 1. 74. Jerarquía de clases del paquete javax.microedition.lcdui

Autor: Byron Delpino

- **Clase Display**

Representa el manejador de la pantalla. Todo MIDlet debe poseer al menos un objeto Display, esta clase es accedida a través del método getDisplay() realizada dentro del constructor del MIDlet.

Un Display se declara:

Display pantalla= Display.getDisplay(this)

- **Clase Displayable**

La clase Displayable representa la(s) pantalla(s) de la aplicación, desplegada(s) a través del método setCurrent. **pantalla.setCurrent(Formulario).**

- **Clase Command**

Esta clase representa los botones que se visualizan en la pantalla. Un botón es implementado cuando se quiere ejecutar una acción o evento.

Existen tres parámetros que se deben definir cuando se declara un objeto Command.

- **Etiqueta:** Cadena de texto que aparecerá en la pantalla.
- **Tipo:** Tipo de objeto Command:
  - **BACK:** Envía al usuario a la pantalla anterior.
  - **CANCEL:** Cancela una acción.
  - **EXIT:** Sale de la aplicación.
  - **OK:** Aceptación de una acción por parte del usuario.
  - **STOP:** Detiene una operación.

- **Prioridad:** Orden de aparición del Command dentro de la pantalla, a mayor número menor prioridad.

Un Command se declara:

Private Command nombre= new Command(String Etiqueta, Command.TIPO , int Prioridad)

- **Clase CommandListener**

Una vez que se ha creado un Command, se debe implementar la interfaz CommandListener, en donde a través del método **commandAction(Command c, Displayable d)** se indica la acción a realizarse.

- **Clase Screen**

La clase Screen es una superclase que contiene todas las clases que incorporar campos de texto, alertas, listas y formularios.

- **Clase Form**

Un formulario (clase Form) es un contenedor de un número de elementos: imágenes, campos de texto, listas, etiquetas.

Todos los objetos dentro de la clase Item deben estar dentro de un formulario mediante el método append().

Un formulario se crea: **Prívate Form formulario= new Form (String título)**

- **Clase TextBox**

Utilizado para la inserción de texto dentro de una caja de texto. La clase TextBox admite el uso de Restricciones para validar el texto ingresado dentro de la caja de texto.

Valor	Definición
ANY	Cualquier carácter.
NUMERIC	Solo números.
PASSWORD	Oculta los caracteres introducidos ***.

Tabla 1. 34. Restricciones de entrada de caracteres

Autor: Byron Delpino

Un TextBox se declara de la siguiente forma:

TextBox (String titulo, String texto, int tamaño, restricciones)

Métodos

Método	Definición
String getString()	Devuelve el contenido del TextBox.
setString(String texto)	Establece el contenido del TextBox.
int size()	Devuelve el número de caracteres

Tabla 1. 35. Métodos de la clase List

Autor: Byron Delpino

- **Clase List**

Esta clase permite construir pantallas que proporcionen una lista de opciones.

Existen tres tipos de listas:

- **IMPLICIT:** Lista en que la selección de un elemento provoca un evento. La acción a ejecutar se la implementa a través del método **commandAction(Command c, Displayable d)** .
- **EXCLUSIVE:** Lista en que un solo elemento puede ser seleccionado a la vez.
- **MULTIPLE:** Selección de uno o varios elementos a la vez.

Existen dos constructores utilizados para implementar listas:

List(String Titulo, Tipo)

List (String Título, Tipo, String[] elementos, Image[] Imágenes)

Métodos

Método	Definición
insert(int posición, String texto, Image imagen)	Inserta un elemento en una posición determinada.
int getSelectedIndex()	Obtiene el índice del elemento seleccionado.
String getString(int posicion)	Obtiene el texto de un elemento determinado por su posición.
Boolean isSelected(int posicion)	Determina si un elemento está seleccionado
int size	Obtiene el número de elementos de una lista

Tabla 1. 36. Métodos de la clase List

Autor: Byron Delpino

- **Clase Item**

La clase Item es una superclase que contiene todas las clases para incorporar etiquetas, imágenes, fechas.

- **Clase TextField**

Es un campo de texto para la edición de texto, es muy parecido a la clase TextBox, con la única diferencia que este debe ser insertado dentro del formulario.

En forma similar que la clase TextBox, TextField admite el uso de Restricciones, para validar el texto ingresado dentro de la caja de texto.

Valor	Definición
ANY	Cualquier carácter.
NUMERIC	Solo números.
PASSWORD	Oculta los caracteres introducidos ***.

Tabla 1. 37. Restricciones de entrada de caracteres

Autor: Byron Delpino

Un TextField se declara de la siguiente forma:

TextField (String titulo, String texto, int tamaño, TextField.Restricciones)

Métodos

Método	Definición
String getString()	Devuelve el contenido del TextField.
setString(String texto)	Establece el contenido del TextField.
int size()	Devuelve el número de caracteres

Tabla 1. 38. Métodos de la clase List

Autor: Byron Delpino

- **Clase StringItem**

Esta clase facilita el despliega de cadena de texto o etiquetas, las cuales no son modificables.

Para crear un StringItem se tiene dos constructores:

StringItem(String etiqueta, String texto)

StringItem(String etiqueta, String texto, StringItem.Apariencia)

Donde:

Etiqueta: Es la etiqueta del item.

Texto: El texto que lo contiene

Apariencia: La apariencia del item, plano(StringItem.PLAIN), enlace (StringItem.HYPERLINK).

Métodos

Método	Definición
String getText()	Devuelve el contenido del StringItem.
setText(String texto)	Establece el contenido del StringItem.

Tabla 1. 39. Métodos de la clase StringItem

Autor: Byron Delpino

- **Clase ImageItem**

La clase ImageItem brinda la posibilidad de incluir imágenes en un formulario.

Para crear un ImageItem se tiene el constructor:

ImageItem(String etiqueta, Image imagen, ImageItem.LAYOUT_Posición, String Alternativo)

Donde:

Etiqueta: Es la etiqueta del item.

LAYOUT: Indica la posición de la imagen en la pantalla. LAYOUT_LEFT (izquierda), LAYOUT_RIGHT (derecha), LAYOUT_CENTER (centro).

Alternativo: Representa el texto que se mostrará en caso de que la imagen exceda la capacidad de la pantalla.

Métodos

Método	Definición
setImage(Image imagen)	Establece una imagen.

Tabla 1. 40. Métodos de la clase ImageItem

Autor: Byron Delpino

- **Clase DateField**

Clase utilizada para el manejo de fechas y horas dentro del formulario.

Para crear un DateField se tiene el constructor:

DateField(String etiqueta, DateField.Modo)

Donde:

Etiqueta: Es la etiqueta del item.

Modo: Fecha (DATE), hora (TIME), hora y fecha (DATE_TIME).

- **Clase ChoiceGroup**

Es un grupo de elementos que se pueden seleccionar, es muy similar a la clase List, su diferencia radica en que el primero debe ser incluido dentro de una clase Form (Formulario).

Existen dos tipos de ChoiceGroup.

- **EXCLUSIVE:** Se puede elegir un solo elemento a la vez.
- **MULTIPLE:** Selección de varios elementos a la vez.

Existen dos constructores utilizados para implementar listas:

ChoiceGroup(String Titulo, ChoiceGroup.TIPO)

ChoiceGroup (String Título, Tipo, String[] elementos, Image[] Imágenes)

Métodos

Método	Definición
insert(int posición, String texto, Image imagen)	Inserta un elemento en una posición determinada.
int getSelectedIndex()	Obtiene el índice del elemento seleccionado.
String getString(int posicion)	Obtiene el texto de un elemento determinado por su posición.
Boolean isSelected(int posicion)	Determina si un elemento está seleccionado
int size	Obtiene el número de elementos de una lista

Tabla 1. 41. Métodos de la clase ChoiceGroup

Autor: Byron Delpino

1.22. Comunicación HTTP

HTTP es un protocolo de tipo petición- respuesta, donde el cliente solicita algún recurso o información y el servidor es quien responde.

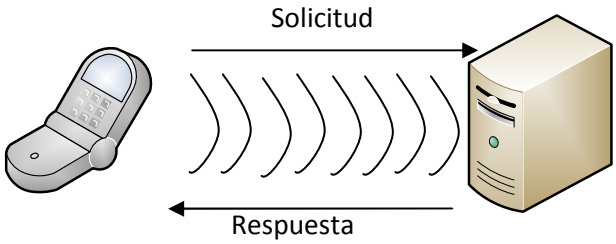


Figura 1. 75. Comunicación cliente-servidor HTTP

Autor: Byron Delpino

Para una comunicación HTTP en J2ME, el dispositivo móvil (MIDlet) cumple el rol de cliente y el servidor a través de un servlet responde a las solicitudes planteadas.

1.22.1. Paquete javax.microedition.io

Para el manejo de conexiones de red, J2ME utiliza los paquetes **javax.microedition.io** y **java.io**. El primer paquete contiene numerosas clases que permiten el manejo y la creación de diferentes conexiones de red, mientras que java.io proporciona las clases (InputStream, OutputSteam) para leer y escribir en dichas conexiones.

- **Clase Connector**

La clase Connector es empleada para abrir una comunicación cliente-servidor a través del método open(). **Connector.open(URL)**

- **Clase Connection**

Esta clases es la encarga de cerrar una conexión HTTP mediante el método close().

- **Interfaz InputConnection**

Empleada para la lectura de un flujo de datos de entrada (InputStream) mediante el método openInputStream(). **InputStream Nombre= conexion.openDataInputStream()**

- **Interfaz OutputConnection**

OutputConnection permite la escritura de un flujo de datos de salida (OutputStream) a través del método openInputStream().

OutputStream Nombre= conexion.openDataOutputStream()

- **Interfaz HttpConnection**

HttpConnection implementa una serie de métodos útiles para crear una conexión HTTP.

Métodos

Método	Definición
setRequestMethod (Tipo de Petición)	Establece el tipo de petición
int getResponseCode	Devuelve el código de Estado

Tabla 1. 42. Métodos de la clase HttpConnection

Autor: Byron Delpino

1.22.2. Estados HTTP

La comunicación HTTP, tanto el cliente como el servidor intercambian paquetes de datos. Básicamente un paquete está conformado por una cabecera, que contiene informacion como direcciones IP del cliente y servidor, puerto, URL, e/o y el cuerpo del paquete que representa los datos en sí.

Una conexión HTTP pasa por tres estados:

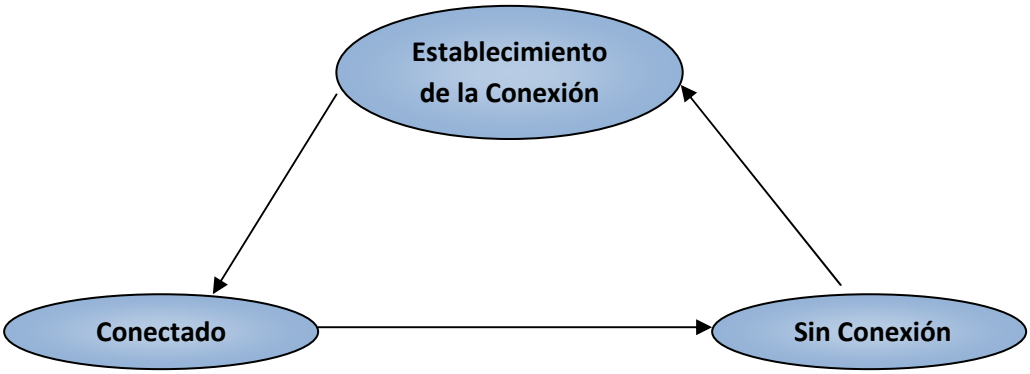


Figura 1. 76. Estados de una conexión HTTP

Autor: Byron Delpino

- **Establecimiento de la conexión**: Se establecen parámetros de comunicación como idioma, formato, tipo de petición.

  Existen 3 tipos de petición:

  - **GET**: Petición de información en donde los datos se envían como parte de la URL.

    Ejm:

    http//:localhost:8080/USB/Index.html?nombre=Byron&edad=22

    URL

    Dato1 nombre Dato2 edad

    Protocolo: HTTP
    IP del Servidor: localhost,
    Puerto: 8080
    Directorio del Recurso: USB
    Recurso: Index.html

    Para la separación entre la URL y los datos de usuario se utiliza el caracter '**?**' , mientras que para la concatenación de datos se emplea **'&'.**

- **POST**: Petición de Información en donde los datos se envían aparte en un stream (flujo).

  Ejm:

  URL: http//:localhost:8080/USB/Index.html

  Datos: nombre=Byron, edad=22, nota=20

- **HEAD:** Petición de información de cabecera.

El establecimiento del tipo de petición dentro del MIDlet se origina utilizando el método **setRequestMethod(Tipo de Petición)**.

- **Estado de Conexión**: Representa el intercambio de información entre cliente-servidor. Una vez que el cliente ha enviado la solicitud al servidor, puede o no haber una respuesta por parte de este, para identificar el estado de respuesta se utiliza el método **getResponseCode().**

  El método getResponseCode utiliza algunos códigos que permiten identificar si hubo o no respuesta.

- **Sin conexión**: Fin de la comunicación entre el cliente y servidor.

1.22.3. Servlets

La comunicación HTTP en J2ME se da entre un dispositivo móvil (MIDLET) y un servidor que almacena un Servlet. Un Servlet es una pequeña aplicación que se ejecuta dentro de un navegador web.

Para la creación de Servlets se requiere el uso del paquete javax.servlet y la implementación de uno de los siguientes métodos:

Método	Definición
doGet (HttpServletRequest request, HttpServletResponse response)	Método que responde a una petición tipo GET.
doPost (HttpServletRequest request, HttpServletResponse response)	Método que responde a una petición tipo POST.

Tabla 1. 43. Métodos a implementarse en un Servlet

Autor: Byron Delpino

Tanto doGet como doPost requieren de los parámetros tipo HttpServletRequest y HttpServletResponse, para la manipulación de los datos que provienen o van hacia el cliente.

1.23. Introducción a Android

Android es un sistema operativo libre de Google orientado para la creación de aplicaciones en equipos móviles. Fue creado en el 2005 por Android.Inc- Google bajo el sistema operativo Linux, su licencia es gratuita.

En la actualidad un sinnúmero de empresas como Dell, Intel, Motorola o Samsung han desarrollado dispositivos que utilizan Android.

Figura 1. 77. Logo de Android

Fuente: es.wikipedia.org/wiki/Android

Además de ser libre y de código abierto, Android presenta las siguientes características:

- Utiliza la máquina virtual Dalvik, para interpretar y ejecutar el código java.

- Permite la representación de gráficos 2D-3D.
- Emplea SQlite para el manejo de base de datos.
- Soporta tecnologías GSM (Global System Position), Wi-Fi, Bluetooth, Wimax, Bluetooth.
- Posee soporte multimedia MPEG-4, MP3, JPEG, PNG, 3GP, e/o.
- Permite videollamadas a través de Google Talk.

El sistema operativo Android está conformado por:

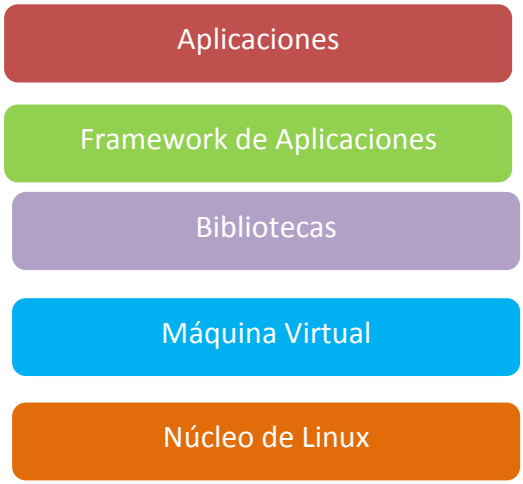


Figura 1. 78. Arquitectura del S.O. Android

Autor: Byron Delpino

- **Aplicaciones**
- **Framework** para el desarrollo de aplicaciones y acceso a las APIs.
- **Bibliotecas:** Android incluye un conjunto de librerías bajo C/C++. Ejm: SQLite.
- **Máquina Virtual:** Dalvik, encargado de ejecutar código java en formato .dex. Esta máquina virtual está pensada para un menor consumo de procesamiento, memoria y energía del teléfono.
- **Núcleo Linux:** Android depende de Linux para servicios de seguridad, gestión de memoria, procesos.

Android además de las APIs ofrecidas por java (java.math, java.net, java.text e/o) establece un sinnúmero de paquetes para el desarrollo de aplicaciones, entre las más importantes se encuentran:

- **android.bluetooth**: Clases que proporcionan la funcionalidad del bluetooth.
- **android.content**: Clases para el acceso y publicación de datos en un dispositivo.
- **android.database**: Contiene clases para explorar datos devueltos a través de un provedor de contenido.
- **android.graphics**: API para la creación de gráficos de bajo nivel.
- **android.hardware**: Proporciona clases para el manejo de cámaras, sensores y otros elementos del dispositivo.
- **android.net**: Clases para acceder a la red.
- **android.text**: Proporciona clases para la creación y edición de texto en pantalla.
- **android.util**: API para manipulación hora/fecha, cadenas, técnicas de conversión.
- **android.view**: Interfaz de usuario.

1.23.1. Descarga e Instalación

El SDK (Software Developer Kid) facilita el compile y ejecución de aplicaciones. EL SDK de Android contiene un depurador, bibliotecas, un emulador (para emular las aplicaciones en la PC), ejemplos y tutoriales.

- ✓ En la página de Android Developers: http://developer.android.com/sdk/index.html, se puede descargar el SDK.

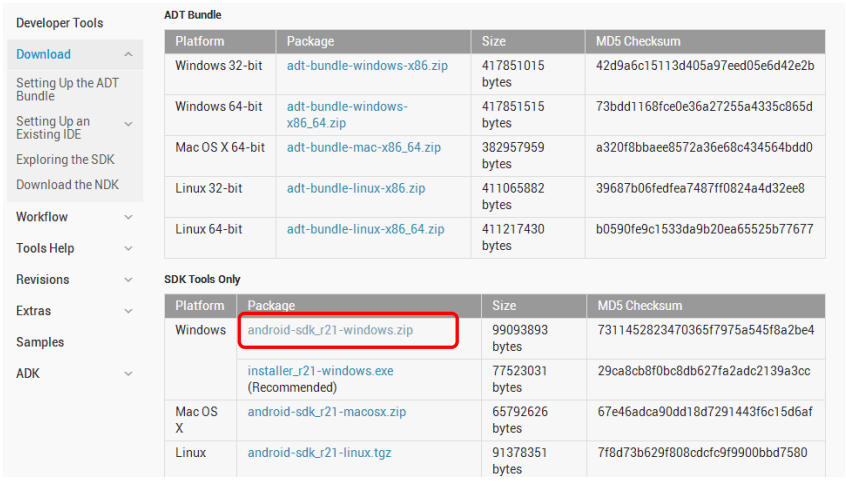


Figura 1. 79. Sitio de descarga del SDK de Android

Fuente: www.developer.android.com/

- ✓ Descargado el archivo en formato zip. se lo descomprime.

✓ A continuación doble clic en **SDK Manager**.

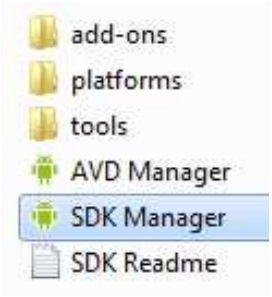


Figura 1. 80. Carpetas y Aplicaciones del SDK de Android- SDK Manager

Autor: Byron Delpino

➢ Se elige los paquetes a instalarse en el equipo.

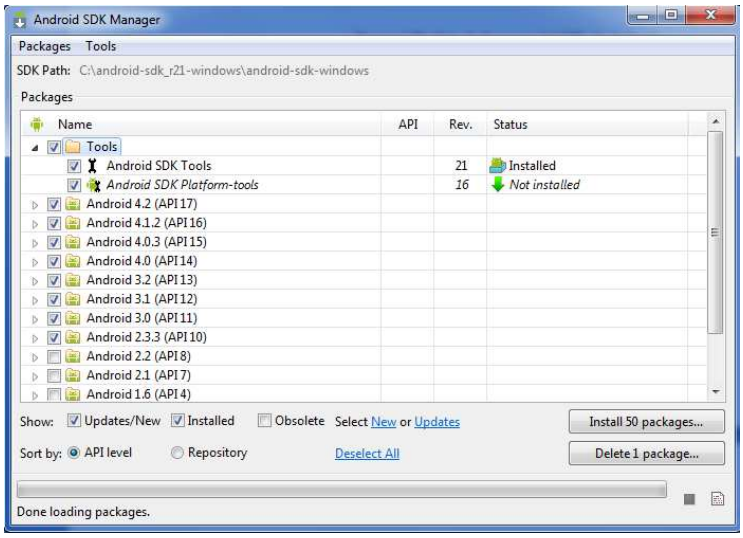


Figura 1. 81. Instalación de paquetes del SDK de Android

Autor: Byron Delpino

➢ Instalado todos los paquetes, se procede a crear un AVD (Android Virtual Device), que permite emular en la pc las aplicaciones Android a desarrollarse.

➢ Se da doble clic en **AVD Manager**.

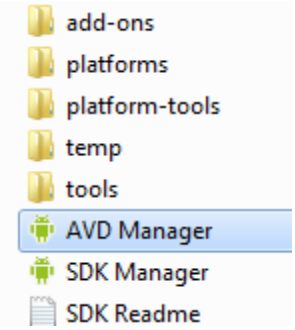


Figura 1. 82. AVD Manager de Android

Autor: Byron Delpino

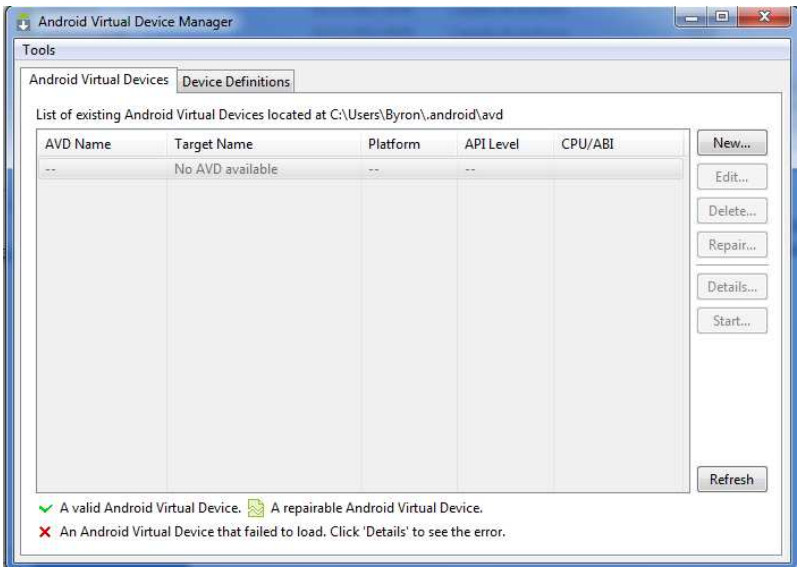


Figura 1. 83. Creación de un dispositivo virtual

Autor: Byron Delpino

- Clic en **New**.
- Se determina parámetros como el nombre del dispositivo virtual, tipo, Memorias, SD. Clic en **OK**.

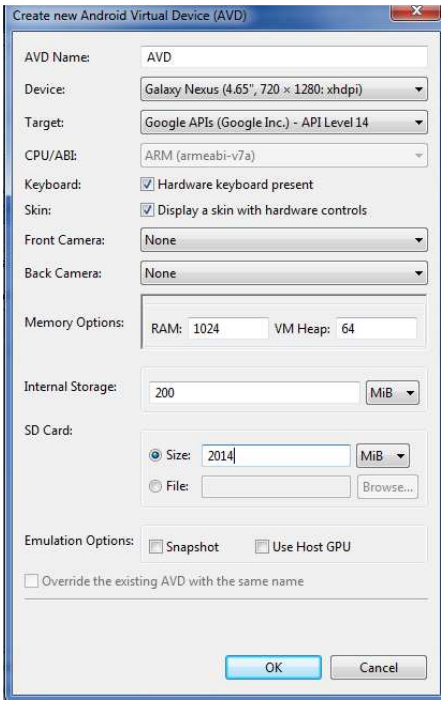


Figura 1. 84. Configuración de parámetros de un AVD

Autor: Byron Delpino

- Se configura Eclipse las Herramientas de Desarrollo de Android (ADT) para la creación, compilación y depuración de aplicaciones Android.
- Arrancar Eclipse, Clic en la pestaña **Help- Install New Software.**

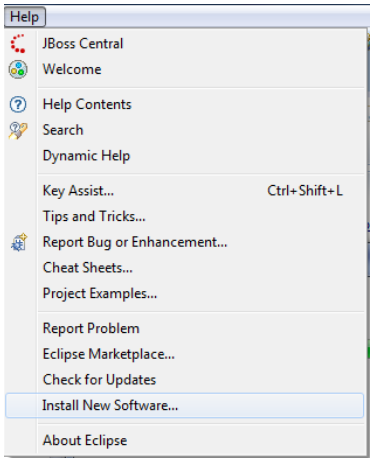


Figura 1. 85. Instalación de las Herramientas de Desarrollo de Android en Eclipse

Autor: Byron Delpino

- En la ventana de instalación digitar en Work with: http://dl-ssl.google.com/android/eclipse/, clic en **Add**.
- Se añade un nombre.

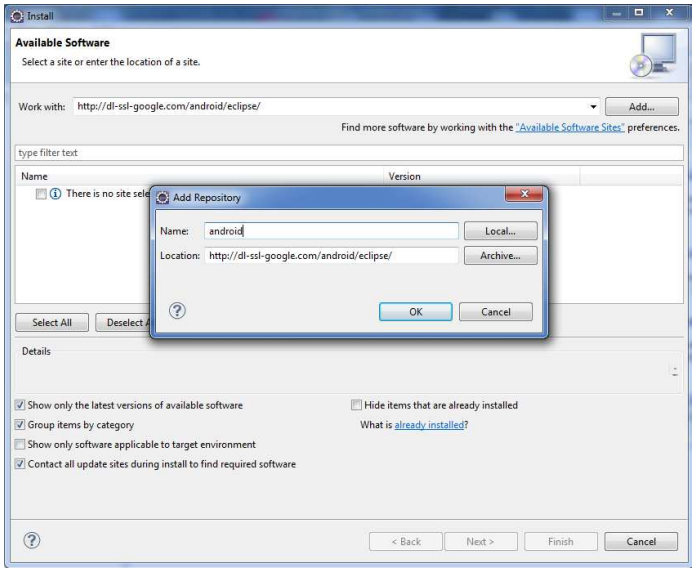


Figura 1. 86. Instalación del ADT en Eclipse

Autor: Byron Delpino

- Se activa la casilla Developer Tools, clic en Next.

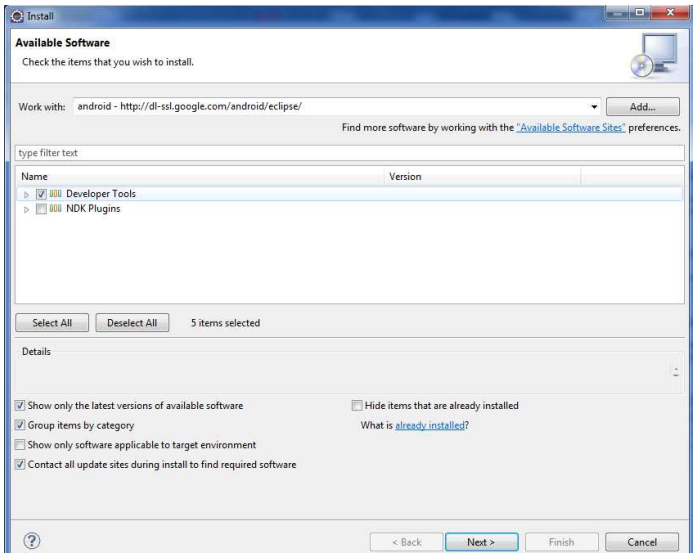


Figura 1. 87. Selección de los paquetes ADT

Autor: Byron Delpino

- Clic **Next**, se observa los detalles de instalación **Next**.
- Se acepta los términos y condiciones de licencia, **Finish.**
- Instalado las herramientas de desarrollo de Android en eclipse, se procede con su configuración.
- Para ello al arrancar el IDE, se despliega una ventana solicitando la configuración del SDK, se da clic en **Use existing SDKs, Next, Finish**.

Figura 1. 88. Configuración SDK en Eclipse

Autor: Byron Delpino

1.24. Laboratorios

1.24.1. Introducción a J2ME

Este laboratorio es un ejemplo básico de una aplicación J2ME, desarrollado para familiarizar al estudiante con la programación en dispositivos móviles, los métodos y constructores que se deben implementar, así como los componentes y librerías requeridas.

1.24.1.2. Diagrama de Bloques

Figura 1. 89. Diagrama de Bloques, Introducción J2ME

Autor: Byron Delpino

1.24.1.3. Programación Java

- En Netbeans, crear una aplicación java, File- New Project – Java ME – Mobile Aplication, Nombre y Ubicación de la aplicación –Configuración: CLDC- 1.1, Perfil MIDP- 2.0 –Finish.
- Se crea el MIDlet.

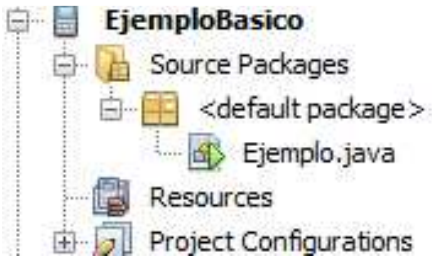


Figura 1. 90. Creación de MIDlet Ejemplo.java

Autor: Byron Delpino

1.24.1.3.1. Definición de Variables

Nombre	Tipo	Definición
pantalla	Display	Componente utilizado para desplegar el formulario.
formulario	Form	Formulario

Tabla 1. 44. Definición de componentes utilizados en Ejemplo.java

Autor: Byron Delpino

1.24.2. Listas Implícitas

Aplicación J2ME implementada para definir el uso de listas implícitas. Se tiene un listado de las facultades de electrónica (Redes, Control y Telecomunicaciones), donde a partir del elemento seleccionado se despliega un formulario correspondiente a cada facultad.

1.24.2.1. Diagrama de Bloques

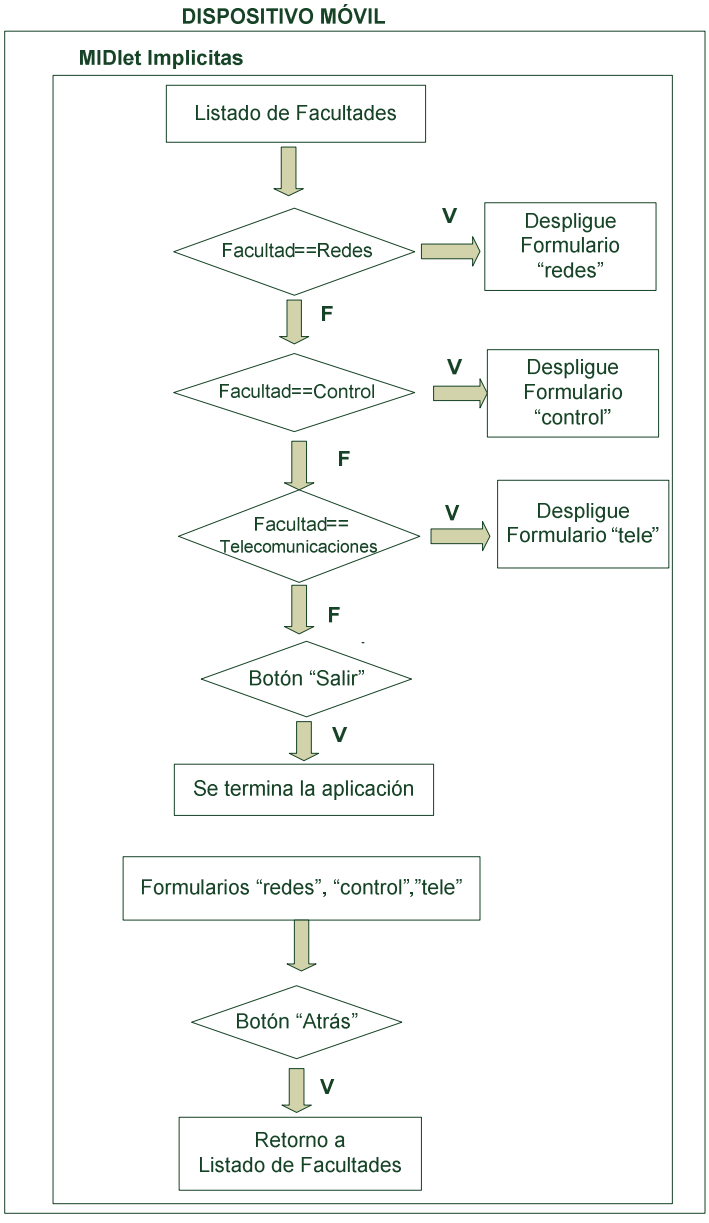


Figura 1. 91. Diagrama de Bloques, Listas Implícitas

Autor: Byron Delpino

1.24.2.2. Programación Java

- En Netbeans, crear una aplicación java, File- New Project – Java ME – Mobile Aplication, Nombre y Ubicación de la aplicación –Configuración: CLDC- 1.1, Perfil MIDP- 2.0 –Finish.
- Se crea el MIDlet.

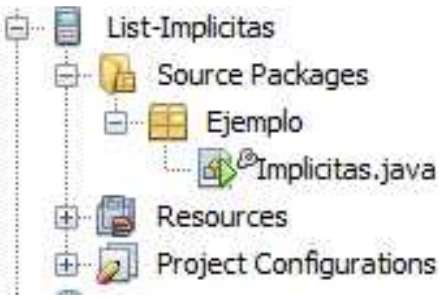


Figura 1. 92. Creación de MIDlet Promedio.java

Autor: Byron Delpino

1.24.2.2.1. Definición de Variables

Nombre	Tipo	Definición
pantalla	Display	Componente utilizado para desplegar formulario.
menu	List	Listado utilizado para elegir facultad.
redes	Form	Formulario de Redes.
redes	Form	Formulario de Control.
redes	Form	Formulario de Telecomunicaciones.
salir	Command	Botón para salir de la aplicación
atras	Command	Botón para ir hacia el listado desde algún formulario (redes, control, tele).
fac	Arreglo de Strings	Variable utilizada para ingresar elementos hacia el Listado menu ("Redes", "Control", "Telecomunicaciones").

Tabla 1. 45. Definición de Variables utilizadas en Implicitas.java

Autor: Byron Delpino

1.24.3. Uso de componentes StringItem, TextField y Command

Aplicación J2ME que consiste en el ingreso de notas de un estudiante en 3 campos de texto (TextField), las cuales a través un Botón (Command), calcula su promedio y lo despliega (StringItem).

1.24.3.1. Diagrama de Bloques

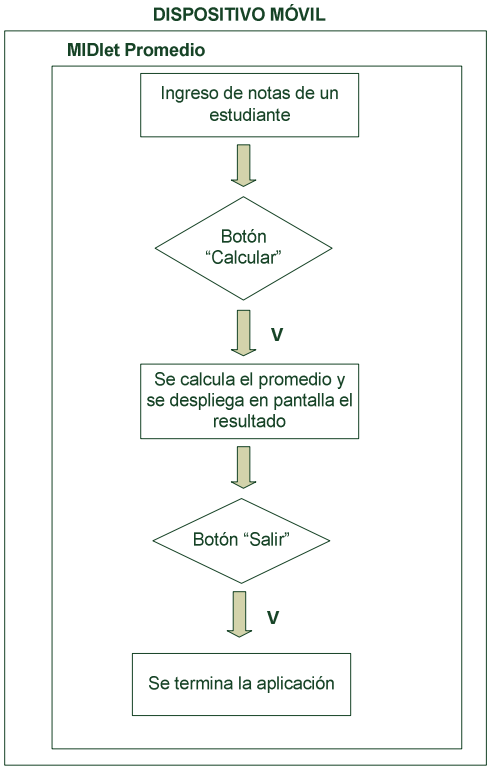


Figura 1. 93. Diagrama de Bloques, Uso de componentes StringItem, TextField y Command

Autor: Byron Delpino

1.24.3.2. Programación Java

- En Netbeans, crear una aplicación java, File- New Project – Java ME – Mobile Aplication, Nombre y Ubicación de la aplicación –Configuración: CLDC- 1.1, Perfil MIDP- 2.0 –Finish.
- Se crea el MIDlet.

Figura 1. 94. Creación de MIDlet Promedio.java

Autor: Byron Delpino

1.24.3.2.1. Definición de Variables

Nombre	Tipo	Definición
pantalla	Display	Componente utilizado para desplegar formularios.
ingreso	Form	Formulario que almacena los StringItems, TextField y Command.
nota1	TextField	Campo de Texto donde se ingresa nota del primer parcial.
nota2	TextField	Campo de Texto donde se ingresa nota del segundo parcial.
nota3	TextField	Campo de Texto donde se ingresa nota del tercer parcial.
resultado	StringItem	Etiqueta que despliega el promedio
calcular	Command	Botón que calcular el promedio de las 3 notas
salir	Command	Botón para salir de la aplicación
promedio	Doble	Variable auxiliar utilizada para calcular el promedio

Tabla 1. 46. Definición de Variables utilizadas en Promedio.java

Autor: Byron Delpino

1.24.4. ChoiceGroup

Este laboratorio utiliza la clase ChoiceGroup de tipo exclusiva y múltiple. El usuario elige la facultad a la que pertenece y los idiomas que habla.

1.24.4.1. Diagrama de Bloques

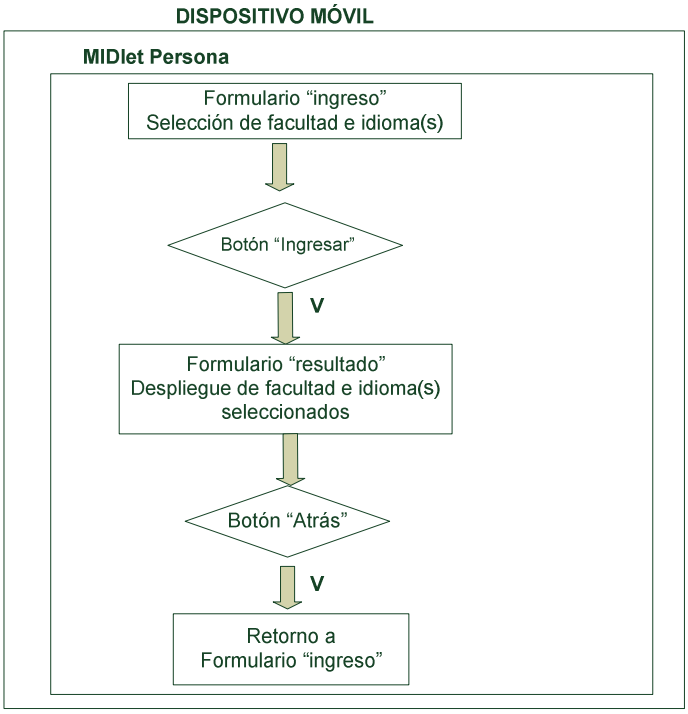


Figura 1. 95. Diagrama de Bloques, ChoiceGroup

Autor: Byron Delpino

1.24.4.2. Programación Java

- En Netbeans, crear una aplicación java, File- New Project – Java ME – Mobile Aplication, Nombre y Ubicación de la aplicación –Configuración: CLDC- 1.1, Perfil MIDP- 2.0 –Finish.
- Se crea el MIDlet.

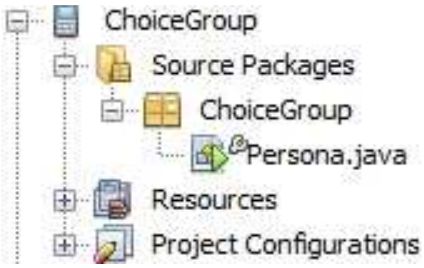


Figura 1. 96. Creación de MIDlet Persona.java

Autor: Byron Delpino

1.24.4.2.1. Definición de Variables

Nombre	Tipo	Definición
pantalla	Display	Componente utilizado para desplegar formularios.
ingreso	Form	Formulario para selección de datos.
resultado	Form	Formulario para desplegar datos seleccionados
facultad	ChoiceGroup	Listado utilizado para elegir facultad
idiomas	ChoiceGroup	Listado utilizado para elegir idioma(s)
ingresar	Command	Botón utilizado para llamar a formulario resultado y desplegar datos ingresados
atras	Command	Botón para ir hacia el formulario “ingreso” desde “resultado”
datoFacultad	StringItem	Etiqueta que despliega la facultad seleccionada
datoIdiomas	StringItem	Etiqueta que despliega los idiomas seleccionados
fac	Arreglo de Strings	Variable utilizada para ingresar elementos hacia el Listado facultad ("Redes", "Control", "Telecomunicaciones").
idio	Arreglo de Strings	Variable utilizada para ingresar elementos hacia el Listado idiomas

		("Inglés", "Francés", "Alemán", "Quichua").
posicion	Entero	Variable auxiliar empleada para obtener el valor del elemento seleccionado del listado facultad
facultadelegida	String	Variable auxiliar que almacena el valor del elemento seleccionado dentro de facultad.
idiomasseleccionados	String	Variable auxiliar que almacena los valores de los elementos seleccionados dentro de idiomas.
Seleccionados	Arreglo de Booleanos	Variable que almacena true o false, dependiendo de si seleccionó o no un elemento dentro de idiomas.

Tabla 1. 47. Definición de Variables utilizadas en Persona.java

Autor: Byron Delpino

1.24.5. Uso de Componentes J2ME

Aplicación J2ME que emplea componentes como StringItem, TextField, ChoiceGroup, ImageItem y DateField. El usuario inserta nombre, facultad, nota y fecha de evaluación, una vez que se presiona el botón “Ingresar”, se despliega los datos ingresados.

1.24.5.1. Diagrama de Bloques

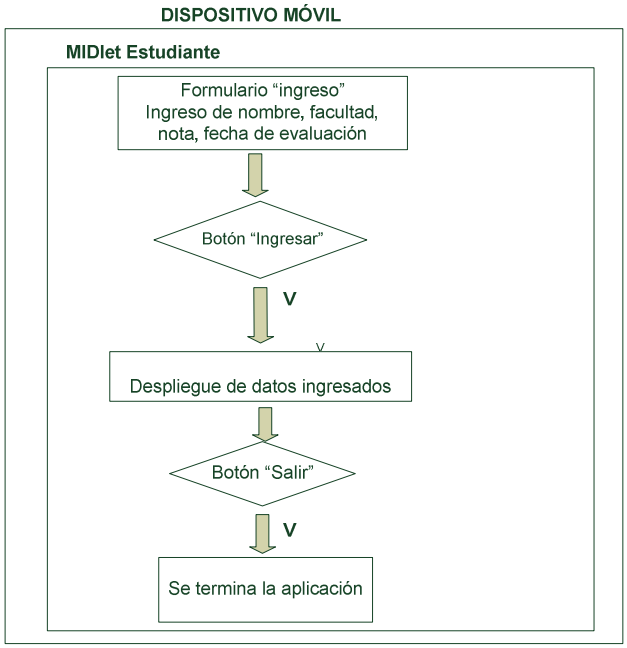


Figura 1. 97. Diagrama de Bloques, Componentes J2ME

Autor: Byron Delpino

1.24.5.2. Programación Java

- En Netbeans, crear una aplicación java, File- New Project – Java ME – Mobile Aplication, Nombre y Ubicación de la aplicación –Configuración: CLDC- 1.1, Perfil MIDP- 2.0 –Finish.
- Se crea el MIDlet.

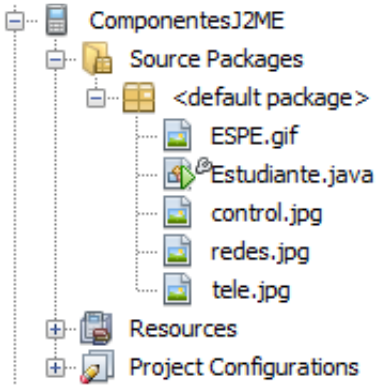


Figura 1. 98. Creación de MIDlet Estudiante.java

Autor: Byron Delpino

1.24.5.2.1. Definición de Variables

Nombre	Tipo	Definición
pantalla	Display	Componente utilizado para desplegar el formulario “ingreso”.
ingreso	Form	Formulario para selección de datos.
titulo	StringItem	Etiqueta que despliega el título de la aplicación
imagen	ImageItem	Imagen desplegada en el formulario
nombre	TextField	Campo de Texto donde se ingresa el nombre del estudiante
facultad	ChoiceGroup	Listado utilizado para elegir facultad
nota	TextField	Campo de Texto donde se ingresa nota del estudiante
fechae	DateField	Componente empleada para seleccionar fecha de evaluación.
nombrei	StringItem	Etiqueta que despliega el nombre ingresado
facultadi	StringItem	Etiqueta que despliega la facultad elegida
notai	StringItem	Etiqueta que despliega la nota ingresada
fechai	StringItem	Etiqueta que despliega la fecha de evaluación ingresada
ingresar	Command	Botón utilizado para desplegar datos ingresados
salir	Command	Botón empleado para salir de la aplicación
fac	Arreglo de Strings	Variable utilizada para ingresar elementos hacia el Listado facultad ("Redes", "Control", "Telecomunicaciones")
facultades	Arreglo de Image	Almacena logos de cada una de las facultades
espe	Image	Variable que importa imagen ESPE
posicion	Entero	Variable auxiliar empleada para obtener el valor del elemento seleccionado del listado facultad
facultadelegida	String	Variable auxiliar que almacena el valor del elemento seleccionado dentro de facultad.

Tabla 1. 48. Definición de Variables utilizadas en Estudiante.java

Autor: Byron Delpino

1.24.6. Comunicación Midlet-Servlet

Este laboratorio está orientado a establecer una comunicación HTTP entre el cliente (MIDlet) y el servidor a través de un Servlet. El usuario envía un dato correspondiente a la facultad, y el servidor responde con información relacionada a redes, control o telecomunicaciones.

1.24.6.1. Diagrama de Bloques

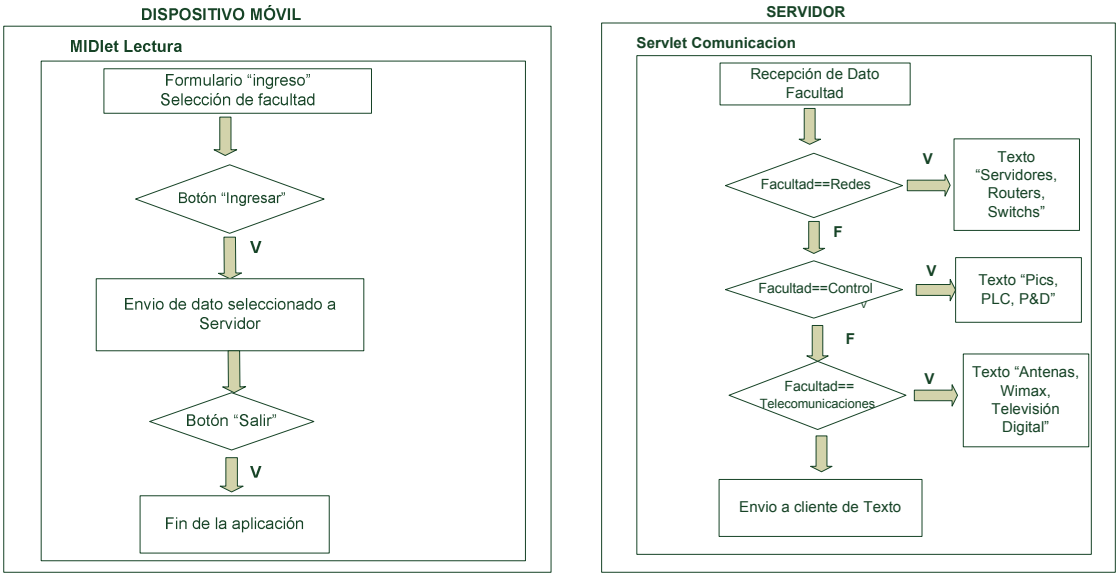


Figura 1. 99. Diagrama de Bloques, Comunicación MIDlet-Servlet

Autor: Byron Delpino

1.24.6.2. Programación Java -MIDlet

- En Netbeans, crear una aplicación java, File- New Project – Java ME – Mobile Aplication, Nombre y Ubicación de la aplicación –Configuración: CLDC- 1.1, Perfil MIDP- 2.0 –Finish.
- Se crea el MIDlet.

Figura 1. 100. Creación de MIDlet Lectura.java

Autor: Byron Delpino

1.24.6.2.1. Definición de Variables

Nombre	Tipo	Definición
pantalla	Display	Componente utilizado para desplegar el formulario “ingreso”.
ingreso	Form	Formulario para selección de datos.
facultad	ChoiceGroup	Listado utilizado para elegir facultad
leer	Command	Botón para enviar a servidor facultad seleccionada
salir	Command	Botón empleado para salir de la aplicación
fac	Arreglo de Strings	Variable utilizada para ingresar elementos hacia el Listado facultad ("Redes", "Control", "Telecomunicaciones")
posicion	Entero	Variable auxiliar empleada para obtener el valor del elemento seleccionado del listado facultad
facultadelegida	String	Variable auxiliar que almacena el valor del elemento seleccionado dentro de facultad.
resultado	StringItem	Etiqueta empleada para desplegar dato recibido desde servidor
hc	HttpConnection	Establece una conexión http
url	String	Url requerida para establecer la comunicación http
is	InputStream	Flujo de datos de entrada que recibe datos desde servidor

respuesta	String	Variable auxiliar útil para obtener valor de dato recibido (arreglo datos)
datos	Arreglo de bytes	Almacena los datos receptados
tamaño	Entero	Longitud de arreglo de bytes datos

Tabla 1. 49. Definición de Variables utilizadas en Lectura.java

Autor: Byron Delpino

1.24.6.3. Programación Java -Servlet

- En Netbeans, crear una aplicación java, New File – Java Web – Web Aplication, Nombre y Ubicación de la aplicación- selección de Servidor GlassFish y JEE 6 Web —Finish.

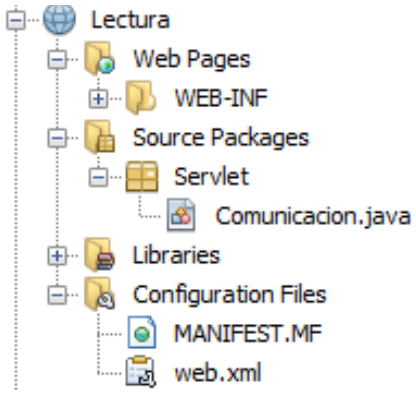


Figura 1. 101. Creación de ficheros Comunicación.java

Autor: Byron Delpino

1.24.6.3.1. Definición de Variables

Nombre	Tipo	Definición
nombre	String	Variable para obtener datos que el cliente emitió
Res1	res1	Variable para envio de datos al cliente
out	PrintWriter	PrintWriter para el envio de datos

Tabla 1. 50. Definición de Variables utilizadas en Servlet Comunicacion.java

Autor: Byron Delpino

1.24.7. Integración J2ME-JSF: Envio y Recepción de Datos

Este laboratorio está orientado a establecer una comunicación HTTP entre el cliente (MIDlet) y el servidor a través de un Servlet. Tanto el cliente como el servidor envían y reciben datos.

1.24.7.1. Diagrama de Bloques

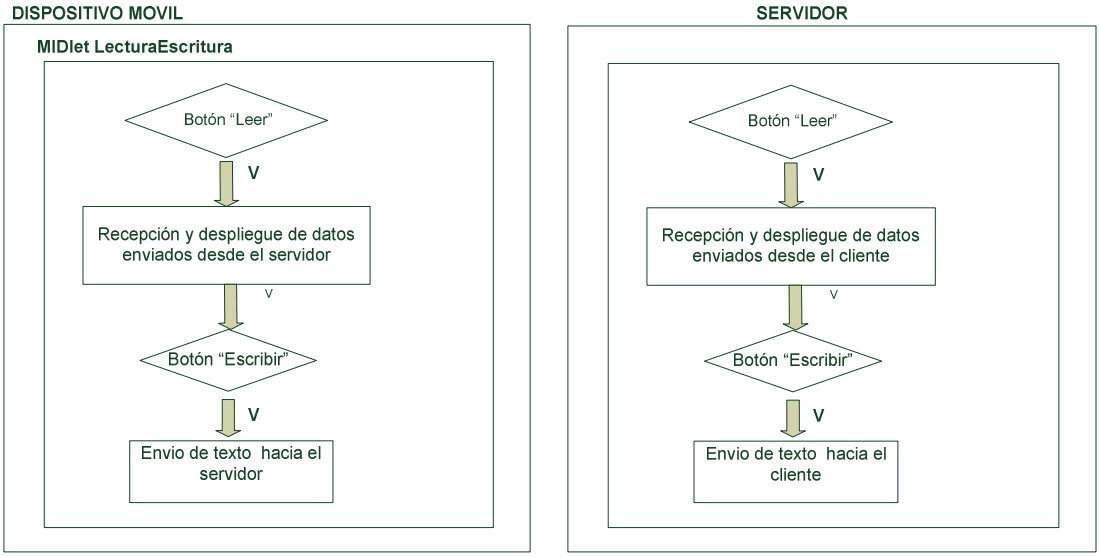


Figura 1. 102. Diagrama de Bloques, Integración J2ME-JSF

Autor: Byron Delpino

1.24.7.2. Programación Java -MIDlet

- En Netbeans, crear una aplicación java, File- New Project – Java ME – Mobile Aplication, Nombre y Ubicación de la aplicación –Configuración: CLDC- 1.1, Perfil MIDP- 2.0 –Finish.
- Se crea el MIDlet.

Figura 1. 103. Creación de MIDlet LecturaEscritura.java

Autor: Byron Delpino

1.24.7.2.1. Definición de Variables

Nombre	Tipo	Definición
pantalla	Display	Componente utilizado para desplegar el formulario "ingreso".
ingreso	Form	Formulario para selección de datos.
envio	TextField	Texto a enviar a servidor
resultado	StringItem	Etiqueta empleada para desplegar texto recibido desde servidor
leer	Command	Botón para recibir texto desde servidor
escribir	Command	Botón para enviar a servidor texto ingresado
hc	HttpConnection	Establece una conexión http
url	String	Url requerida para establecer la comunicación http
is	InputStream	Flujo de datos de entrada que recibe datos desde servidor
respuesta	String	Variable auxiliar útil para obtener valor de dato recibido (arreglo datos)
datos	Arreglo de bytes	Almacena los datos receptados
tamaño	Entero	Longitud de arreglo de bytes datos

Tabla 1. 51. Definición de Variables utilizadas en LecturaEscritura .java

Autor: Byron Delpino

1.24.7.3. Programación Servlet y JSF

- En Netbeans, crear una aplicación java, New File – Java Web – Web Aplication, Nombre y Ubicación de la aplicación- selección de Servidor GlassFish y JEE 6 Web –Framework: JavaServer Faces –Finish.
- Se crea el archivo .xhtml (Index), .java (pc y Comunicacion) y .xml(face-config).

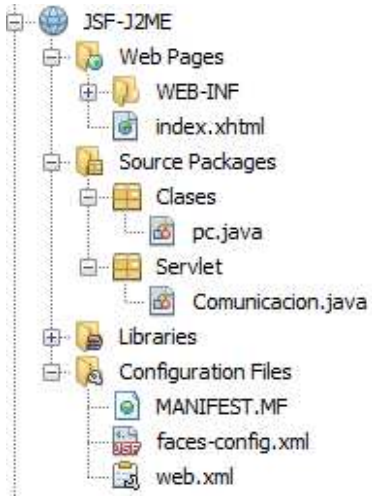


Figura 1. 104. Creación de ficheros xhtml, java y xml

Autor: Byron Delpino

1.24.7.3.1. Definición de Variables

Nombre	Tipo	Definición
com1	Comunicacion	Objeto que permite el enlace entre el servlet y JSF
entrada	String	Datos recibidos desde el cliente
salida	String	Datos enviados al cliente

Tabla 1. 52. Definición de Variables utilizadas en pc.java

Autor: Byron Delpino

Nombre	Tipo	Definición
PC	pc	Objeto que permite el enlace entre el servlet y JSF
textoEnviar	String	Datos a ser enviados al cliente
textoRecibir	String	Datos enviados desde el cliente
out	PrintWriter	PrintWriter para el envio de datos

Tabla 1. 53. Definición de Variables utilizadas en Servlet Comunicacion.java

Autor: Byron Delpino

1.24.8. Integración J2ME-JSF- USB: Envio y Recepción de Datos

Aplicación que establece una comunicación HTTP entre el cliente (MIDlet) y el servidor a través de un Servlet, quienes podrán controlar la lectura y escritura del puerto USB.

La programación USB, así como JSF ha sido revisada en los laboratorios **1.6.2 y 1.17.5**, por lo que para el desarrollo del presente laboratorio únicamente se hará hincapié en la programación del MIDlet y del Servlet.

1.24.8.1. Diagrama de Bloques

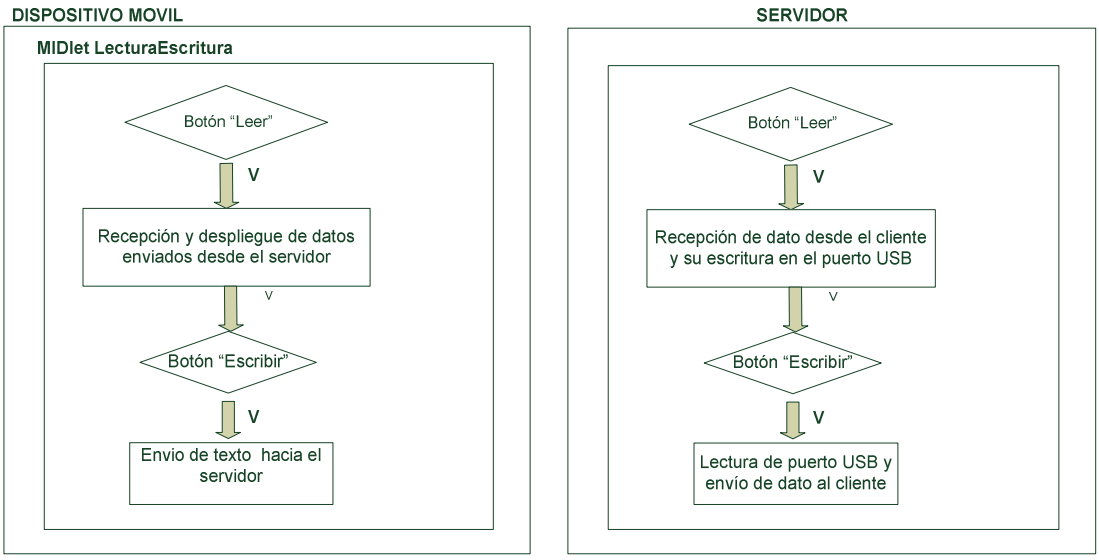


Figura 1. 105. Diagrama de Bloques, Integración J2ME-JSF-USB

Autor: Byron Delpino

1.24.8.2. Programación Java -MIDlet

- En Netbeans, crear una aplicación java, File- New Project – Java ME – Mobile Aplication, Nombre y Ubicación de la aplicación –Configuración: CLDC- 1.1, Perfil MIDP- 2.0 –Finish.
- Se crea el MIDlet.

Figura 1. 106. Creación de MIDlet LecturaEscritura.java

Autor: Byron Delpino

1.24.8.2.1. Definición de Variables

Nombre	Tipo	Definición
pantalla	Display	Componente utilizado para desplegar el formulario “ingreso”.
ingreso	Form	Formulario para selección de datos.
envio	Choicegroup	Grupo de elementos que permite seleccionar dato a enviar
recepcion	StringItem	Etiqueta empleada para desplegar texto recibido desde servidor
leer	Command	Botón para recibir texto desde servidor
escribir	Command	Botón para enviar a servidor texto ingresado
hc	HttpConnection	Establece una conexión http
datos	Arreglo de String	Variable utilizada para ingresar elementos hacia el Listado envio ("1","2","4","8","16",

		"32","64","128")
url	String	Url requerida para establecer la comunicación http
is	InputStream	Flujo de datos de entrada que recibe datos desde servidor
respuesta	String	Variable auxiliar útil para obtener valor de dato recibido (arreglo datos)
datos	Arreglo de bytes	Almacena los datos receptados
tamaño	Entero	Longitud de arreglo de bytes datos

Tabla 1. 54. Definición de Variables utilizadas en LecturaEscritura .java

Autor: Byron Delpino

1.24.8.3. Programación Servlet y JSF

- En Netbeans, crear una aplicación java, New File – Java Web – Web Aplication, Nombre y Ubicación de la aplicación- selección de Servidor GlassFish y JEE 6 Web –Framework: JavaServer Faces –Finish.
- Se crea el archivo .xhtml (Index), .java (pc y Comunicacion) y .xml(face-config).

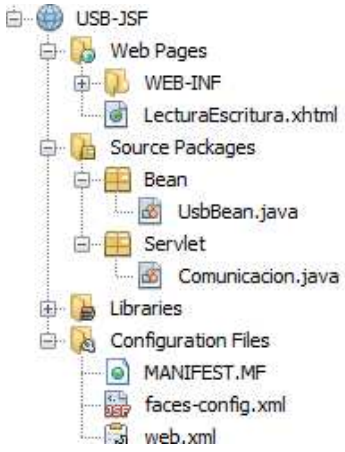


Figura 1. 107. Creación de ficheros xhtml, java y xml

Autor: Byron Delpino

1.24.8.3.1. Definición de Variables

Nombre	Tipo	Definición
recepcion	String	Variable que representa el atributo recepcion del Bean usb
envio	Arreglo de Strings	Variable que representa el atributo envio del Bean usb
lectura	Byte	Comando para Lectura de datos USB
escritura	Byte	Comando para Escritura de datos USB
out	Arreglo de Bytes	Envía la variable lectura o escritura a través del Puerto USB, además envía la variable DATO
VALOR	Byte	Dato a ser enviado por el puerto USB
dato	String	Dato recibido desde puerto USB
a	Entero	Variable auxiliar para obtener el dato a ser escrito en el puerto USB

Tabla 1. 55. Definición de Variables utilizadas en UsbBean.java

Autor: Byron Delpino

Nombre	Tipo	Definición
UsbBean	pc	Objeto que permite el enlace entre el servlet y JSF (Bean)
textoEnviar	String	Datos a ser enviados al cliente
textoRecibir	String	Datos enviados desde el cliente
out	PrintWriter	PrintWriter para el envio de datos

Tabla 1. 56. Definición de Variables utilizadas en Servlet Comunicacion.java

Autor: Byron Delpino

1.24.9. Integración J2ME-JSF- USB: Envio y Recepción de Datos

Aplicación que establece una comunicación HTTP entre el cliente (MIDlet) y el servidor a través de un Servlet, quienes podrán controlar la lectura y escritura del puerto USB (sensor de temperatura y ventiladores).

La programación USB, así como JSF ha sido revisada en los laboratorios **1.6.3 y 1.17.6**, por lo que para el desarrollo del presente laboratorio únicamente se hará hincapié en la programación del MIDlet y del Servlet.

1.24.9.1. Diagrama de Bloques

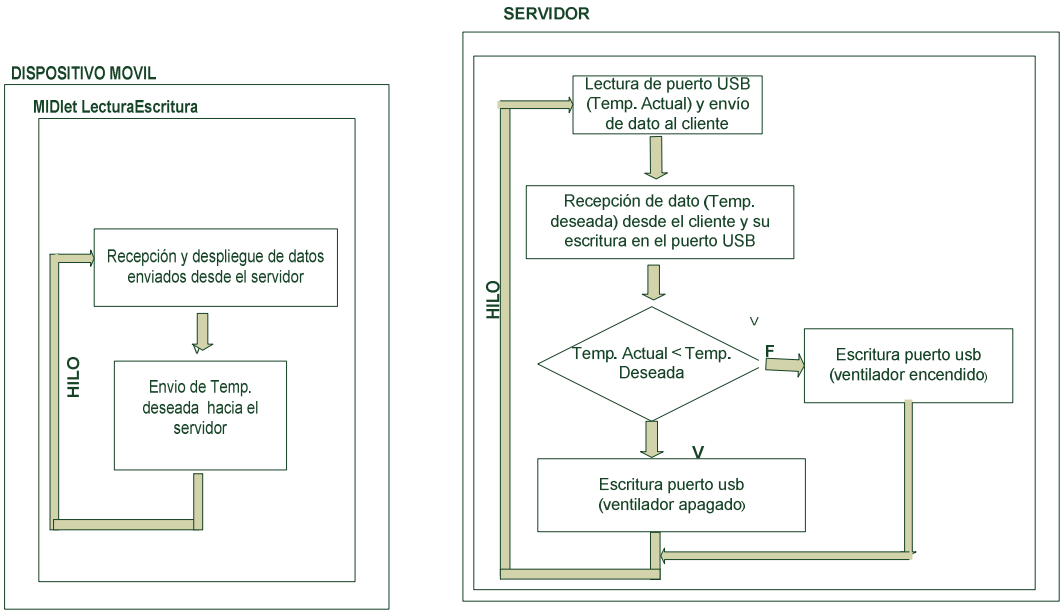


Figura 1. 108. Diagrama de Bloques, Integración J2ME-JSF-USB

Autor: Byron Delpino

1.24.9.2. Programación Java -MIDlet

- En Netbeans, crear una aplicación java, File- New Project – Java ME – Mobile Aplication, Nombre y Ubicación de la aplicación –Configuración: CLDC- 1.1, Perfil MIDP- 2.0 –Finish.
- Se crea el MIDlet.

Figura 1. 109. Creación de MIDlet LecturaEscritura.java

Autor: Byron Delpino

1.24.9.2.1. Definición de Variables

Nombre	Tipo	Definición
pantalla	Display	Componente utilizado para desplegar el formulario "ingreso".
ingreso	Form	Formulario para selección de datos.
envio	Choicegroup	Grupo de elementos que permite seleccionar dato a enviar
recepcion	StringItem	Etiqueta empleada para desplegar texto recibido desde servidor
hc	HttpConnection	Establece una conexión http
datos	Arreglo de String	Variable utilizada para ingresar elementos hacia el Listado envio ("1","2","4","8","16", "32","64","128")
url	String	Url requerida para establecer la comunicación http
is	InputStream	Flujo de datos de entrada que recibe datos desde servidor
respuesta	String	Variable auxiliar útil para obtener valor de dato recibido (arreglo datos)
datos	Arreglo de bytes	Almacena los datos receptados
tamaño	Entero	Longitud de arreglo de bytes datos
H	Hilo	Implementación de clase hilo
T	Thread	Clase Hilo hereda la clase Thread

Tabla 1. 57. Definición de Variables utilizadas en LecturaEscritura .java

Autor: Byron Delpino

1.24.9.3. Programación Servlet y JSF

- En Netbeans, crear una aplicación java, New File – Java Web – Web Aplication, Nombre y Ubicación de la aplicación- selección de Servidor GlassFish y JEE 6 Web –Framework: JavaServer Faces –Finish.
- Se crea el archivo .xhtml (Index), .java (pc y Comunicacion) y .xml(face-config).

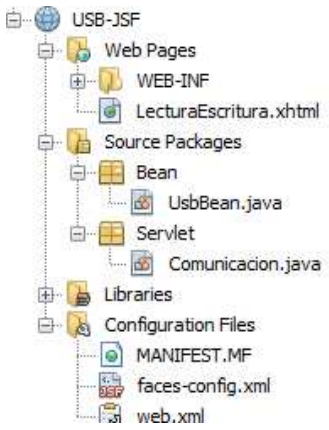


Figura 1. 110. Creación de ficheros xhtml, java y xml

Autor: Byron Delpino

1.24.9.3.1. Definición de Variables

Nombre	Tipo	Definición
recepcion	String	Variable que representa el atributo recepcion del Bean usb
seteo	String	Variable que representa el atributo seteo del Bean usb
ventilador	String	Variable que representa el atributo ventilador del Bean usb
obtener	MeterGaugeChartModel	Variable utilizada para el manejo del componente primefaces medidor de temperatura
intervals	List<Number>	Variable usada para incorporar los intervalos de temperatura en el componente meterGaugeChart

temperatura	Doble	Variable auxiliar utilizada para obtener el valor de la temperatura actual del sensor
Nota	Doble	Variable auxiliar utilizada para establecer el valor del atributo nota
Recepcion	String	Varíable auxiliar empleada para establecer el valor del atributo recepción
Seteo	String	Variable auxiliar utilizada para establecer el valor del atributo seteo
Ventilador	String	Variable auxiliar empleada para establecer el valor del atributo ventilador
VIDyPID	String	VID y PID del PIC
instancia	Entero	Instancia
lectura	Byte	Comando para Lectura de datos USB
escritura	Byte	Comando para Escritura de datos USB
out	Arreglo de Bytes	Envía la variable lectura o escritura a través del Puerto USB, además el valor a enviar por puerto USB
dato	String	Variable auxiliar para recibir dato leído desde el puerto usb
valseteo	String	Variable auxiliar empleada para obtener el valor de la temperatura deseada

Tabla 1. 58. Definición de Variables utilizadas en USBLm35.java

Autor: Byron Delpino

Nombre	Tipo	Definición
USBLm35	pc	Objeto que permite el enlace entre el servlet y JSF (Bean)
textoEnviar	String	Datos a ser enviados al cliente
textoRecibir	String	Datos enviados desde el cliente
out	PrintWriter	PrintWriter para el envio de datos

Tabla 1. 59. Definición de Variables utilizadas en Servlet Comunicacion.java

Autor: Byron Delpino

1.24.10. Introducción a Android

Este laboratorio es un ejemplo básico de una aplicación en Android donde se realiza el despliegue de un mensaje de texto en pantalla.

1.24.10.1. Diagrama de Bloques

Figura 1. 111. Diagrama de Bloques, Introducción Android

Autor: Byron Delpino

1.24.10.2. Programación Java

- En Eclipse, File – New –Project – Android Application Project – Nombre y Ubicación de la aplicación- selección del SDK y Compilador – 4 Next – Finish.

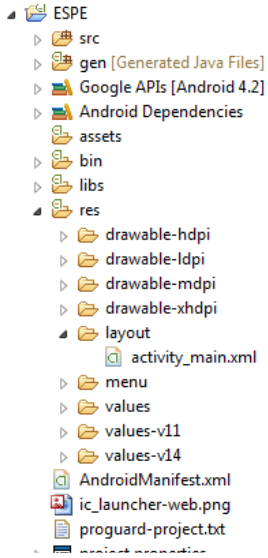


Figura 1. 112. Creación de Aplicación Android

Autor: Byron Delpino

CAPÍTULO 2

DISEÑO E IMPLEMENTACIÓN DE LA PLATAFORMA

2.1. TICS

Las Tecnologías de la Información y Comunicación son un grupo de herramientas que facilitan el tratamiento y transmisión de la información para la construcción del aprendizaje.

Las Tics permiten la integración de nuevas tecnologías para socializar el conocimiento, de esta forma la educación emplea los actuales procesos de enseñanza y aprendizaje.

2.1.1. TECNOLOGÍAS

Son un conjunto de recursos y conocimientos técnicos que permiten el diseño de un bien o servicio que satisface las necesidades del usuario y manipula correctamente la información.

- **Tecnología Informática:** Herramientas que almacenan y procesan informacion a través de sistemas computaciones y software electrónico.

- **Computador:** Equipo electrónico requerido para el procesamiento, almacenamiento y presentación electrónica de datos.

- **Servidor:** Computador central de una red con grandes capacidades y una arquitectura especializada que forma un eje de comunicaciones con n estaciones u otros servidores, para brindar o distribuir diferentes servicios a varios clientes.

- **Red de computadoras:** Es la interconexión entre computadoras para compartir información, recursos y servicios a través de un medio. Existen tres categorías:
  - ✓ **Redes LAN (Red de Área Local):** Grupo de computadoras que pertenecen a una misma organización o zona geográfica.

- ✓ **Redes MAN (Red de Área Metropolitana):** Conexión a alta velocidad de diversas LAN cercanas geográficamente.
- ✓ **Redes WAN (Red de Área Extensa):** Conexión de múltiples LAN y MAN distribuidas en grandes distancias geográficas.

- **Modelo OSI:** Es un modelo de referencia creado por la Organización Estándar Internacional ISO, que define el orden en que todos los sistemas y componentes de una red transmiten datos.

- **Arquitectura TCP/IP:** Describe un conjunto de reglas y protocolos que permiten a un equipo comunicarse dentro de una red. Los protocolos del modelo TCP/IP establecen las normas como los datos deben ser formateados, direccionados, transmitidos y enrutados.

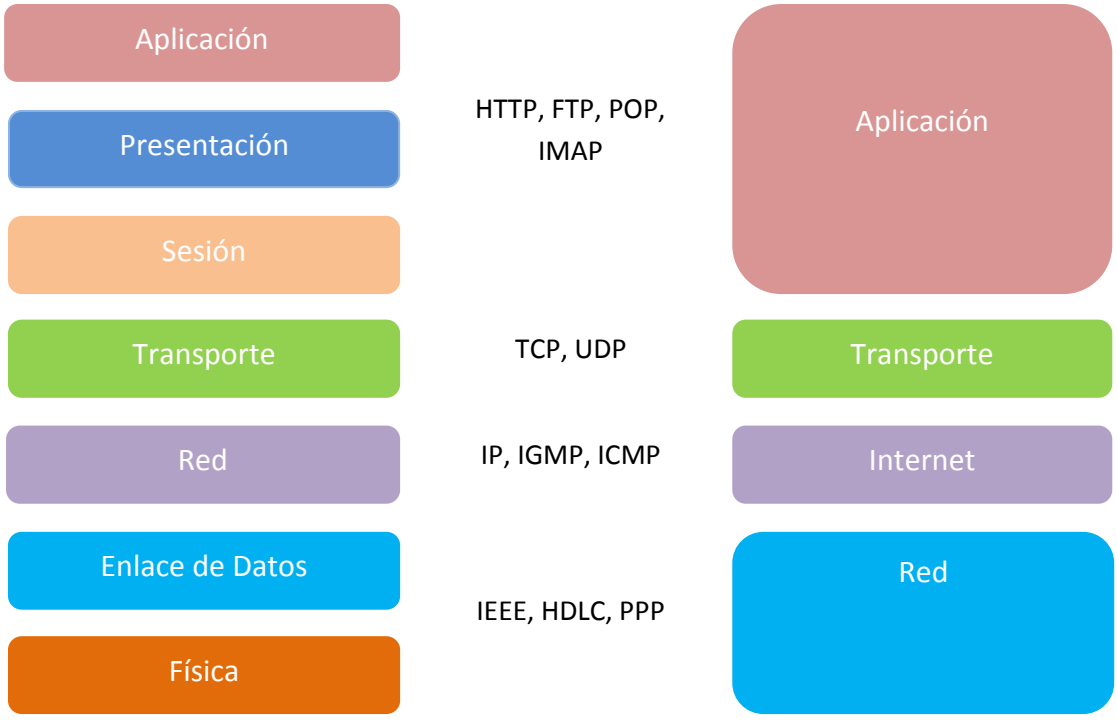


Figura 2. 1. Modelo OSI – Arquitectura TCP/IP

Autor: Byron Delpino

- **Internet:** Grupo de redes interconectadas que utilizan la arquitectura TCP/IP para compartir datos. Mediante el internet se implementa servicios de información como páginas web, correo electrónico, videoconferencias, aprendizaje virtual.

Categoría	Servicio	Puerto	Descripción
Navegación	http	80	Visualización de sitios web.
Servidores Información	ftp tftp	20/21 69	Transferencia de archivos.
Servidores correo	Smtp Pop3 Imap	25 110 143	Envio (Smtp) y Recepción (Pop3, Imap) de correo electrónico.
Sistema de Nombres de Dominio	Dns	53	Resolución de Nombres de Dominio.
Configuración Dinámica de Hosts	Dhcp	67	Asignación de parámetros ip (dirección, máscara, gateways) a estaciones de la red.
Resolución de Direcciones	Arp	-	Se determina la dirección física (MAC) a partir la dirección lógica (IP).
Acceso Remoto	Telnet Ssh	23 22	Acceso remoto a hosts de la red.
Control y Notificación	Icmp	-	Control y Notificación de Errores del Protocolo de Internet. Ejm: ping, tracert.

Tabla 2. 1. Familia de Protocolos de Internet

Autor: Byron Delpino

- **Tecnología Inalámbrica:** Interconexión entre varios equipos a la red sin un medio de propagación físico (cables).
  - **Dispositivo Móvil:** Son equipos con limitadas capacidades computacionales (memoria, procesamiento, interfaz gráfica), empleados para la realización de una o varias tareas en cualquier sitio.
  - **Telefonía Móvil:** Abarca un conjunto de conocimientos y herramientas que permiten la transmisión de datos a través de dispositivos móviles. La

telefonía móvil está conformada por una red de comunicaciones (antenas, repetidoras) y los equipos terminales (móvil).

- ✓ **Primera Generación (1G):** Surgida en los 80, consistía en la transmisión de señales de voz analógicas.
- ✓ **Segunda Generación (2G):** Transmisión digital de la voz, Servicio de Mensajes Cortos (SMS).
- ✓ **Tercera Generación (3G):** Capacidad de transmitir voz y datos. Facilidad para descarga de archivos, visualización de videos, conexión a internet.
- ✓ **Cuarta Generación (4G):** Tecnología aún en desarrollo, orientada a implementar mayores prestaciones como la incorporación de televisión digital, internet a gran velocidad (1Gbps), servicios multimedia High Definition (HD). Red de comunicaciones implementadas sobre IP.

- **Web:** Es un sistema de transmisión de información (texto, gráficos, objetos multimedia) a través del internet. La observación de contenidos se lo realiza mediante páginas web.
  - **Web 1.0:** Sitios estáticos (solo lectura) en donde el usuario no puede interactuar con el contenido de la página.
  - **Web 2.0:** Sitios web enfocados a compartir información de forma dinámica entre todos los usuarios de una red. La Web 2.0 permite la interacción entre un sitio y el usuario a través de aplicaciones como blogs, wikis, redes sociales, foros virtuales.
  - **Web 3.0:** Sitios web dinámicos donde la búsqueda de información se acoge al lenguaje y necesidades del usuario (web semántica). Observación de contenidos y recursos multimedia en 3D en todo tipo de dispositivos (Pcs, móviles).

- **Tecnología Educativa:** Es el desarrollo de metodologías y aplicaciones apoyadas en las Tics que buscan resolver problemas relacionados con la enseñanza y el aprendizaje. Una plataforma tecnológica educativa abarca un amplio rango de aplicaciones informáticas que facilitan a un docente la creación, administración y

distribución de cursos virtuales o aprendizaje electrónico (e-learning) a través de internet.

2.1.2. INFORMACIÓN

Es la organización y procesamiento de un conjunto de datos que constituye un mensaje, instrucción u operación.

- **Tipo de Información**
    - Pública: Cualquier persona tiene libertad para acceder a este tipo de información
    - Privada: Información de acceso restringido.

    La información puede ser distribuida mediante texto, sonido, imágenes, recursos multimedia, e/o. En la informática cada tipo de información tiene formatos o estructuras para grabar los datos dentro de un fichero.
    - **Texto**: .doc, .txt.
    - **Imágenes**: .jpg, .bmp, .png.
    - **Audio**: .wav, .mp3, .wma.
    - **Video**: .mp4, .3gp, .mpeg.

- **Sistemas de Información**: Son todos los elementos (datos, personas, actividades) que interactúan entre sí para el procesamiento, almacenamiento y distribución de la información.

- **Arquitectura de Información:** Es el estudio de estructuras organizativas de información cómo se organiza la información a fin de que el usuario genere conocimiento.

- **Informática:** Conjunto de técnicas y conocimientos que facilitan el tratamiento automático de la información utilizando al computador como principal herramienta.

2.1.3. COMUNICACIÓN

Es el proceso de transmitir información desde un emisor hacia un receptor.

- **Elementos:** Emisor, Mensaje, Canal (Medio), Receptor, Código (Lenguaje o Idioma).

- **Medios de Comunicación:** Representa el instrumento por el cual se transmite el mensaje (Libros, Radio, TV, Internet).

- **Telecomunicaciones:** Es una técnica empleada para transmitir todo tipo de datos (señales, imágenes, voz, texto) a través de cables, medios ópticos, físicos o electromagnéticos. Las telecomunicaciones abarcan:
    - **Tratamiento de Señales:** Estudio del tratamiento y acondicionamiento de señales, codificación, amplificación, filtraje de ruido, e/o.
    - **Redes Inalámbricas:** Orientado al estudio de las transmisión y recepción de datos mediante ondas electromagnéticas. Ejm: Tecnologías Wimax, Wifi. Satelital.
    - **Telefonía:** Estudio de la transmisión de la voz (señales acústicas) por medio de señales eléctricas.
    - **Fibra Óptica:** Medio empleado para transmitir datos mediante pulsos de luz.

- **Comunicación Educativa:** Consiste en la interacción entre docente y alumno creando un clima de óptimo aprendizaje.

2.1.4. INFORMÁTICA EDUCATIVA

La informática educativa es la integración de la informática en el plano educativo para promover el aprendizaje a través del computador.

Ventajas

- ✓ Interacción entre alumno y computador. El computador permite que el estudiante participe activamente en el proceso de aprendizaje.

- ✓ Atención individual al estudiante. La informática educativa se acoge a las necesidades y ritmo de cada alumno.
- ✓ Las herramientas computacionales permiten al estudiante aprender y pensar de forma creativa.
- ✓ La capacidad del computador para el empleo de la evaluación como medio de aprendizaje.
- ✓ La capacidad que otorga al alumno en el control del contenido de aprendizaje.

2.1.5. EDUCACIÓN VIRTUAL Y METODOLOGÍA PACIE

Es el uso de las herramientas que proporcionan las Tics para el control de contenidos y el desarrollo de un aprendizaje que se adapte a los tiempos y necesidades del estudiante.

La educación virtual es aquella forma de estudio que a través de herramientas tecnológicas establece la planificación y guía de un tutor, así como la interrelación profesor-alumno.

En la actualidad, la educación virtual cumple un rol determinante en los centros de enseñanza, puesto que la educación es una forma de integración hacia el mundo tecnológico.

Para la implementación de este tipo de enseñanza y el fortalecimiento de las Tics, se desarrolló la metodología PACIE (Presencia Alcance Capacitación Interacción E-Learning).

PACIE fue creada por el fundador de FATLA (Fundación para la actualización tecnológica de Latinoamérica) Ingeniero Pedro Camacho en el 2004.

PACIE es una metodología para el empleo de herramientas virtuales en todo tipo de educación (Presencial, Semipresencial, Distancia).

- **Presencia:** Emplear todas los mecanismos para que el alumno ingrese a la aula virtual y aproveche de todos los recursos implementados en la misma.
- **Alcance:** Trazar los objetivos que busca el desarrollo de un aula virtual hacia el alumnado.

- **Capacitación:** El docente debe estar capacitado para guiar correctamente al alumno.
- **Interacción:** Desarrollar actividades que estimulen el socializar y compartir conocimiento.
- **E-Learning:** Hacer uso de la tecnología para impartir conocimiento de acuerdo a los estándares pedagógicos.

2.2. FUNDAMENTACIÓN PEDAGÓGICA

Ciencia encargada del estudio de la educación y la enseñanza, la pedagogía recibe influencia de la psicología, la sociología, la antropología, la historia y la medicina.

La pedagogía busca establecer los mejores métodos para fortalecer la educación de un estudiante, además procura que el individuo se convierta en un ser autodidacta, motivado por aprender e investigar.

2.2.1. TEORÍAS DEL APRENDIZAJE

Una teoría de aprendizaje pretende describir los procesos como un sujeto accede al conocimiento. Las teorías de aprendizaje se centran en la adquisición de capacidades para razonar y adquirir conceptos.

TEORÍA	DESCRIPCIÓN
Teorías asociativas, asociacionistas o del condicionamiento	Basada en el esquema estímulo respuesta. El aprendizaje se basa en la repetición.
Teorías cognitivas	Centrada en el razonamiento y en lo mental, buscan una concepción integral de las cosas.
Teorías funcionalistas	Mediante una serie de actividades o funciones dinámicas un individuo se adapta al medio.
Teorías estructuralistas	El aprendizaje es una cadena de procesos interrelacionados dirigidos a la formación de estructuras mentales.

Teorías psicoanalíticas	Se basa en conductas innatas de un sujeto, haciendo hincapié a problemas surgidos en su infancia.
Teorías conductistas	Interpretación de la conducta humana en base al comportamiento animal.
Teorías no directivas	Centran el aprendizaje en las experiencias de un individuo.
Teorías matemáticas, estocásticas	Se basan en la estadística para el análisis de diferentes fenómenos que intervienen en el aprendizaje.
Teorías centradas en los fenómenos o en áreas y clases particulares de comportamiento.	Aprendizaje a través de la práctica, observación, curiosidad, refuerzo, castigo, proceso verbales, entre otras.

Tabla 2. 2. Teorías de Aprendizaje

Autor: Byron Delpino

2.2.2. MÉTODOS Y TÉCNICAS DE ENSEÑANZA

Son un grupo de recursos requeridos para una ordenada, metódica y adecuada enseñanza. Los métodos y técnicas buscan hacen más eficiente el proceso de aprendizaje para la elaboración de conocimientos, la adquisición de habilidades y el cumplimientos de los objetivos educativos. Un método es el camino adecuado para llegar un objetivo, la técnica es la herramienta requerida para cumplirlo.

Métodos:

- **Método Deductivo:** El contenido revisado va de lo general a lo particular. El docente presenta ideas, conceptos generales y el estudiante obtiene conclusiones.
- **Método Inductivo:** Este método va de lo particular a lo general. A partir de hechos o experiencias particulares se llega a conclusiones finales.
- **Método Lógico:** Consiste en el análisis de un tema siguiendo un orden establecido, desde lo simple a lo complejo. Método usado en adolescentes y adultos.

- **Método Psicológico:** El estudio de un elemento se basa en las experiencias y necesidades del alumno, no sigue un orden lógico. Método aplicado en la enseñanza primaria.
- **Método Verbalístico -Simbólico:** El aprendizaje se da mediante la comunicación oral entre docente-alumno y el empleo de gráficos, símbolos tablas.
- **Método Analítico:** Implica la descomposición o separación de un todo en sus partes. Para comprender un fenómeno es necesario conocer los elementos que lo constituyen.
- **Método Sintético:** El estudio de un tema se lo realiza a partir de sus elementos constitutivos, hasta llegar a un todo. Consiste en la reconstrucción o unión de los elementos.

Técnicas de Enseñanza

- Cátedras
- Discurso con materiales de apoyo
- Discusiones
- Preguntas y respuestas
- Demostraciones
- Ejercicios prácticos, Talleres
- Experimentos
- Visitas Técnicas

2.3. FUNDAMENTACIÓN FILOSÓFICA

La filosofía es el estudio de interrogantes asociadas a la existencia, moral, mente, conocimiento y la educación

La filosofía en la educación cumple un papel muy importante en la búsqueda de responder a todas las inquietudes presentadas en el proceso de enseñanza-aprendizaje, así como el pensamiento y concepción de la sociedad acerca de la educación.

La filosofía educativa pretende una comprensión minuciosa sobre el actuar educativo a fin de que el alumno despierte su espíritu investigativo.

Epistemología

La Epistemología es una rama de filosofía, centrada en examinar los problemas relacionados con el conocimiento en general. La Epistemología se enfoca en el análisis y fundamentación de las condiciones de producción y validación del conocimiento científico.

2.4. FUNDAMENTACIÓN PSICOLÓGICA

La psicología es el estudio de la actividad mental y comportamiento de los individuos. La psicología se encarga de analizar aspectos de la mente como el funcionamiento cerebral, la inteligencia, motivación, emoción, personalidad, conciencia, y el inconsciente.

Conductismo: El aprendizaje es un cambio del comportamiento en función de los cambios del entorno. El conductismo manifiesta que el aprendizaje es la asociación de estímulos y respuestas. Esta teoría está enfocada en el aprendizaje de memoria.

Cognitivismo: El aprendizaje se consigue dando una respuesta lógica a la pregunta planteada. El cognitivismo trata de descubrir como la mente humana es capaz de aprender. El cognitivismo a diferencia del conductismo busca el razonamiento en vez de la memorización sin comprensión.

Constructivismo: Esta teoría se enfoca en la construcción de habilidades a partir de un conocimiento previo. El constructivismo es de índole práctico-secuencial debido a que se emplea lo conocido en una situación nueva.

Tanto el cognitivismo como el constructivismo se centran en el aprendizaje a través de la comprensión y el por qué de las cosas y no del memorismo.

2.4.1. TEORÍAS COGNITIVAS DEL APRENDIZAJE

- ✓ **Teoría del Desarrollo cognitivo (Jean Piaget):** El desarrollo cognitivo se da desde la infancia.

- Etapa sensorio-motora (0-2 años): Reconocimiento de objetos ausentes, lenguaje no desarrollado, percepción sensorial.
- Etapa pre-operacional (2-7 años): Desarrollo del lenguaje, pensamiento interiorizado.
- Etapa de Operaciones Concretas (7-11 años): Expresiones más lógicas, pensamiento lógico, socialización.
- Etapa de Operaciones Formales (11 años en adelante): Capacidad de manejar enunciados, pensamiento más allá de la realidad, desarrollo de hipótesis.

- ✓ **Aprendizaje Significativo (David Ausubel):** Valora la experiencia que tiene el aprendiz en su mente. El aprendizaje se da mediante la compresión, transformación y almacenamiento de la información. El estudiante relaciona la información nueva con la ya existente.

- ✓ **Desarrollo cognitivo mediante interacción social (Vygotsky):** El lenguaje y las relaciones sociales influyen en el aprendizaje del infante.

- ✓ **Aprendizaje por Descubrimiento (Jerome Brunner):** Brindar herramientas al estudiante para que participe en la construcción del conocimiento. Orientado a fomentar la investigación.

2.5. MÉTODOS DE ENSEÑANZA

2.5.1. MÉTODOS DE ENSEÑANZA PRESENCIALES

Los métodos de enseñanza presenciales son aquellos en que el proceso de aprendizaje se da mediante la presencia física y comunicación directa entre docente-alumno, basada en la pedagogía de la presencia.

Este tipo de enseñanza se centra en el contacto directo entre los actores y por lo general se desarrolla de manera grupal basada en la cooperación y colaboración, en donde la

presencia del tutor es importante para resolver las dudas planteadas en el aula y bajo la aplicación de diferentes componentes de los procesos educativos.

En la enseñanza presencial se emplea cuadernos, esferográficos, computadoras, retroproyectores como apoyo para generar conocimiento.

2.5.2. MÉTODOS DE ENSEÑANZA A DISTANCIA- VIRTUALES

E-LEARNING

E-Learning (Aprendizaje Electrónico) es un método de educación a distancia completamente virtualizado a través del uso de las Tics (Internet, Plataformas Virtuales), facilitando un aprendizaje interactivo, flexible y accesible.

Características

- Uso de la web 2.0 para el acceso a la información: El tutor y el alumno interactúan mediante blogs, wikis, tareas virtuales, e/o.
- Comunicación estudiante-profesor a distancia de forma sincrónica (videoconferencias, chats) o asincrónica (correo electrónico).
- Aprendizaje flexible apoyado en tutorías y recursos multimedia.
- Almacenamiento de la información sobre un servidor web.
- Para su empleo se requiere un computador, conexión a internet, ofimática.
- Enseñanza individual adaptada a las necesidades y disponibilidad de tiempo de cada alumno.
- Facilita la actualización de la información.
- Requiere conocimientos previos por parte del docente para desarrollar herramientas de aprendizaje adecuadas.
- Se fomenta una cultura de investigación en el alumno.

B-LEARNING

Aprendizaje semi-presencial que utiliza la tecnología como medio de formación académica.

Características

- El alumno es el actor principal en el proceso de aprendizaje.
- Interacción entre docente-estudiante se puede dar con sesiones presenciales.
- Aprendizaje independiente, el alumno debe aprender buscando información relevante en la red.
- Empleo de herramientas tecnológicas (computadora, internet, chat, fotos, correos, mensajería instantánea.).
- El docente crea módulos para que el aprendiz los desarrolle.
- Trabajo en equipo.
- El tutor debe contar con la capacitación para el manejo de nuevas tecnologías.

M-LEARNING

Es el aprendizaje virtual mediante el uso de pequeños dispositivos móviles (celulares, tablets, i-pods) que cuente con alguna forma de conectividad inalámbrica.

Características

- Flexibilidad: Movilidad.
- Disponibilidad: El alumno accede al sitio de aprendizaje cuando cuente con el tiempo necesario.
- Independencia tecnológica: Las aplicaciones virtuales son ejecutadas en cualquier plataforma.
- Navegación sencilla.
- Inserción de recursos multimedia (imágenes, video, audio).
- Presenta problemas debido a las limitaciones de los dispositivos móviles (batería, procesamiento, conectividad).

2.6. SISTEMA DE GESTIÓN DE APRENDIZAJE

Un LMS (Learning Management System) es un software instalado en un servidor web para la creación, administración y control de sitios de formación educativa y programas de entrenamiento.

El objetivo de un LMS es permitir el aprendizaje en cualquier sitio y momento, acogiéndose a las necesidades del estudiante.

Las funciones de un Sistema de Gestión de Aprendizaje son:

- Gestión de usuario, determinación de roles de administrador, docente y alumno.
- Gestión de cursos y grupos.
- Gestión de recursos, contenidos.
- Establecer instrumentos de comunicación entre los actores (wikis, foros, mensajes, audio/videoconferencia).
- Creación de herramientas de evaluación del desempeño del aprendiz mediante exámenes en línea, entrega de tareas.
- Mantener un calendario del curso sobre sus fechas, hitos y actividades.
- Acceso a directorios y bibliotecas online.

RECURSOS Y COMPONENTES DE UN LMS

- **Distribución de contenidos:**
  - ✓ Repositorios virtuales de archivos: para proporcionar espacios para diferentes tipos de archivos y formatos tales como imágenes, video, texto (Biblioteca online), presentaciones.
  - ✓ Inserción de hipervínculos, imágenes, videos.
  - ✓ Edición y Acceso a contenidos en varios formatos (.pdf, .docx, .html).
- **Usuarios: se fundamentan en los diferentes niveles de acción, propiedades, privilegios y roles para cada usuario y sus actividades.**
  - ✓ Administrador
  - ✓ Tutor
  - ✓ Estudiante
- **Herramientas de comunicación y evaluación:**
  - ✓ **Comunicación:** Foros, Correo Electrónico, Chats, Pizarras Virtuales.
  - ✓ **Evaluación:** Creación, edición y corrección de evaluaciones.
- **Herramientas de Administración y Asignación de permisos:**
    - Otorgamiento de permisos y autorizaciones.
    - Asignación de permisos por perfil del usuario (administrador, docente, alumno).

- Proceso de matriculación.

- **Organización:** Representa la distribución de cursos, definición de instancias de aprendizaje para determinados grupos.
- **Herramientas de Seguimiento y Evaluación:**
  - Ficha personal del alumno.
  - Monitoreo y Reporte de actividades del estudiante.

Figura 2. 2. Componentes de una Plataforma E-learning

Fuente: www.e-aula.cl/2011/06/componentes-de-una-plataforma-e-learning/

VERSIONES DE LMS

Hoy en día existe un sinnúmero de LMS disponibles para la comunidad educativa, desde versiones comerciales (WebCT-BlackBoard) así como de libre distribución (Claroline, Moodle, Dokeos).

VERSIONES COMERCIALES

WebCT-BlackBoard

- ✓ Desarrollado en Java Scripts y PHP.
- ✓ Cuenta con un Sistema de Almacenamiento de Archivos.
- ✓ Comunicación sincrónica y asincrónica
- ✓ Monitoreo de avance de los estudiantes

- ✓ Soporte técnico limitado al propietario
- ✓ Versión disponible para equipos móviles

VERSIONES GRATUITAS

Claroline:

- ✓ Desarrollado en PHP.
- ✓ Admite diferentes DBMS (Sistema de Gestión de Base de Datos).
- ✓ Facilidad en el empleo de módulos.
- ✓ Gran comunidad de diseñadores.
- ✓ Interfaz llamativa y amigable con el usuario.

Moodle

- ✓ Plataforma e-learning más utilizada.
- ✓ Desarrollado en PHP
- ✓ Múltiples herramientas para el diseño de cursos.
- ✓ Admite múltiples DBMS
- ✓ Comunicación sincrónica y asincrónica
- ✓ Soporte técnico ilimitado (comunidad de diseñadores).
- ✓ Requiere de cierto grado de capacitación para su administración.

ATutor

- ✓ Desarrollo bajo PHP
- ✓ Interfaz gráfica llamativa
- ✓ Muy utilizado para el diseño de cursos destinados a personas con discapacidad.

2.7. MOODLE

Moodle (Entorno Modular de Aprendizaje Dinámico Orientado a Objetos), es un Sistema de Gestión de Aprendizaje creado por el profesor y científico de computación Martin Dougiamas en el 2002.

Hoy en día Moodle es el LMS de libre licencia más utilizado en el mundo, este software es empleado por instituciones educativas, profesores independientes, empresas privadas, organizaciones.

La primera versión de Moodle fue la 1.0 lanzada al mercado en agosto del 2002, en la actualidad se cuenta con la versión 2.4.1.

Figura 2. 3. Logotipo de Moodle

Fuente: www.moodle.org/?lang=es

Características:

- Emplea la pedagogía constructivista.
- Ideal para clases virtuales.
- Plataforma diseñada bajo PHP.
- Soporte para 91 idiomas.
- Uso de Imap y Pop3 para el servicio de correo electrónico.
- Implementación de políticas de seguridad y acceso.
- Soporta múltiples Gestores de Base de Datos.
- Comunicación Síncrona (Mensajería Instantánea) y Asíncrona (Foros).
- Certificaciones de seguridad TTL (Seguridad de Capa de Transporte) y SSL (Capa de Conexión Segura).

Requerimientos

- **Como Servidor:** Instalación de Apache o IIS (Internet Information Services), y un gestor de base de datos (MySQL, SQL, Oracle).
- **Como Cliente:** Browser y plugins para la visualización de recursos multimedia.

Estructura de una interfaz Moodle

Figura 2. 4. Ventana Principal de un curso virtual creado en Moodle

Autor: Byron Delpino

1. **Barra de Navegación:** Ubicación del sitio dentro de la plataforma.
2. **Barra de Lateral:** Despliegue de actividades, cursos, participantes, administración.
3. **Contenidos**
4. **Activar Edición:** Permite la edición de los contenidos y recursos dentro del curso.
5. **Foros, Noticias.**

2.7.1. RECURSOS Y COMPONENTES DE MOODLE

DISTRIBUCIÓN DE CONTENIDOS

- **Editor HTML:** Inserción de texto, imágenes y videos.

En la plataforma virtual dar clic sobre **Activar Edición** => **Editar Informe**

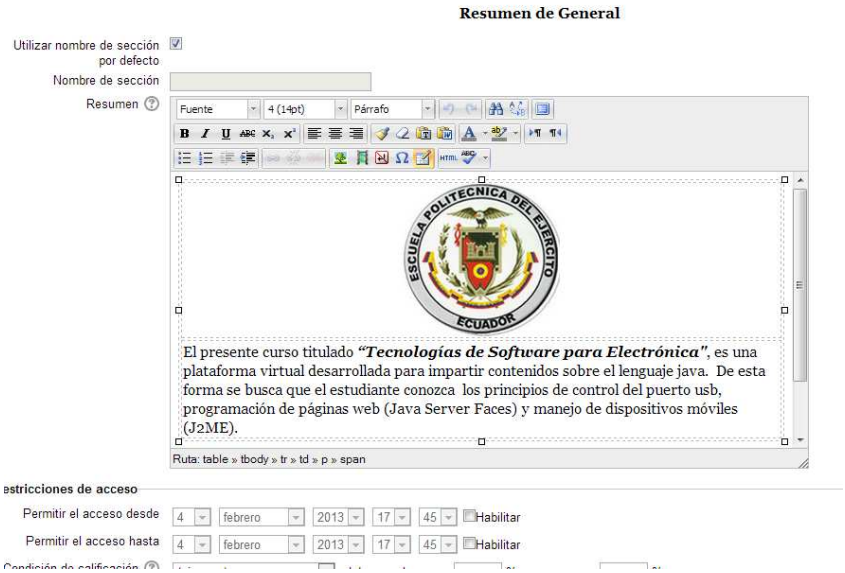


Figura 2. 5. Editor HTML para la inserción de texto, imágenes, video.

Autor: Byron Delpino

El editor HTML permite la inserción y edición (Fuente, Tamaño, Color) de texto, tablas, viñetas.

- **Insertar tablas**

Clic en  , se determina parámetros como el número de filas, columnas, borde, alineación. **Actualizar**.

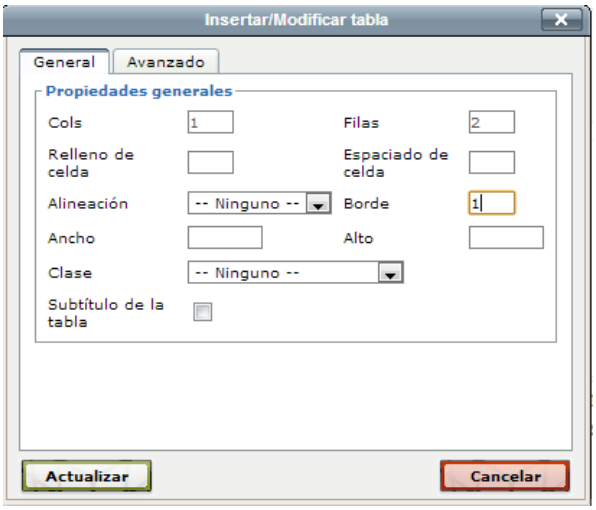


Figura 2. 6. Inserción de tablas en el editor HTML

Autor: Byron Delpino

- **Agregar Imágenes, Videos**

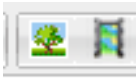
Dar clic sobre uno de estos íconos  (imagen o video), se busca el directorio donde se encuentre el archivo y se procede a cargarlo e insertarlo dentro de la plataforma.

Figura 2. 7. Inserción de imágenes/videos dentro del editor HTML

Autor: Byron Delpino

- **Agregar Etiquetas**

Para insertar etiquetas en la plataforma virtual, clic en **Activar Edición**, **Añadir una Actividad o Recurso, Etiqueta, Agregar.**

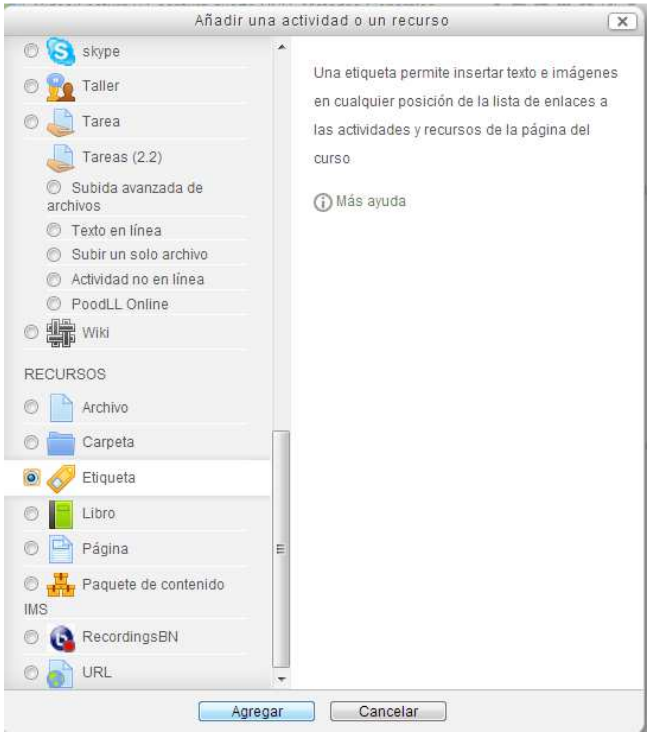


Figura 2. 8. Agregar Etiqueta en la Plataforma

Autor: Byron Delpino

En un editor HTML se añade el texto a desplegar, clic en **Guardar Cambios y Regresar al curso**.

Figura 2. 9. Edición de texto para agregar etiquetas

Autor: Byron Delpino

- **Agregar Archivos**

Para insertar archivos en la plataforma virtual, clic en **Activar Edición**, **Añadir una Actividad o Recurso, Archivo, Agregar.**

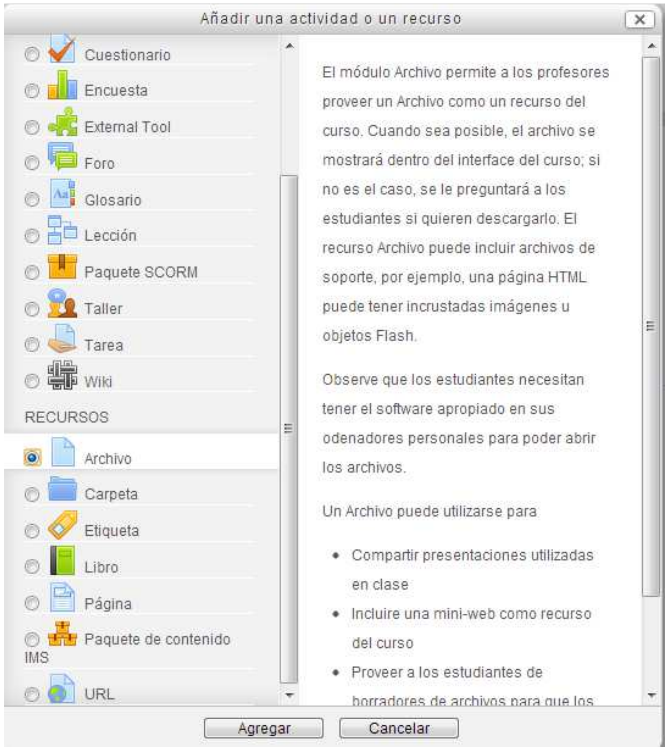


Figura 2. 10. Agregar Archivos en la Plataforma

Autor: Byron Delpino

Se procede a editar el nombre y descripción del recurso.

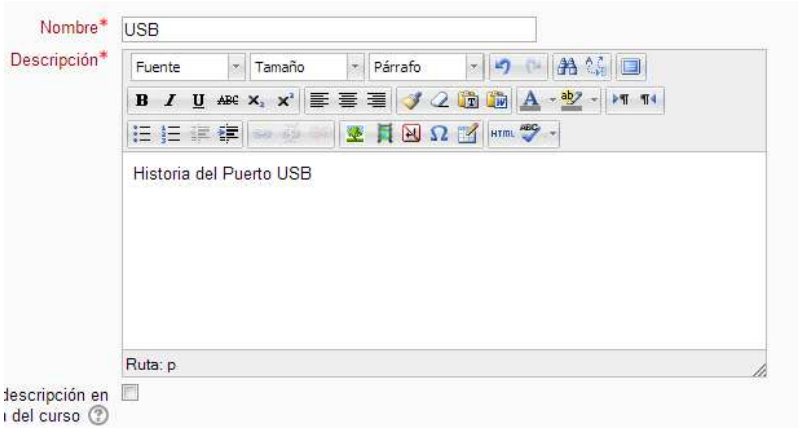


Figura 2. 11. Edición de nombre y descripción del archivo

Autor: Byron Delpino

Se determina el directorio donde se encuentra el archivo, se lo carga en la plataforma, clic en **Guardar Cambios y Regresar al curso**.

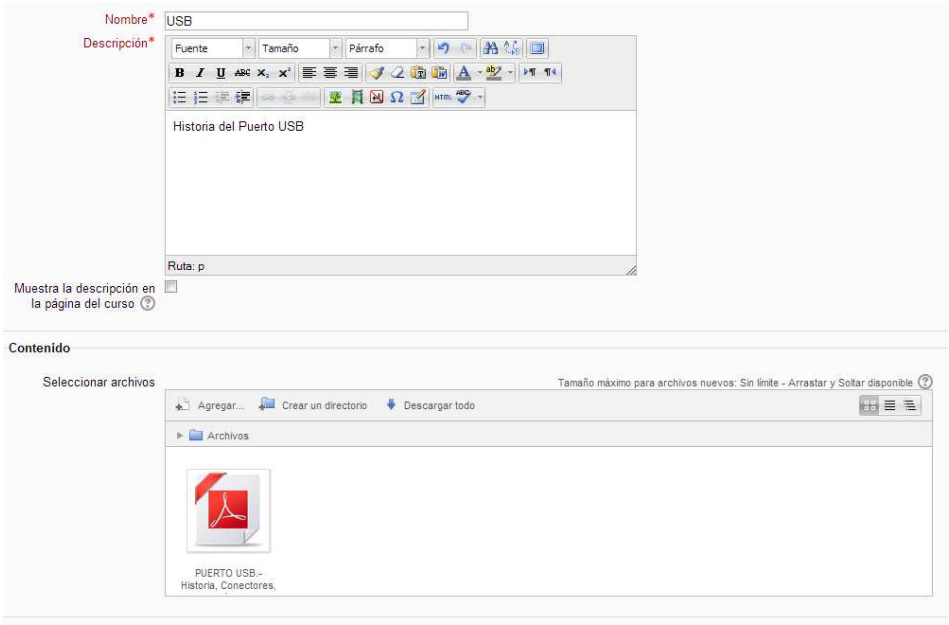


Figura 2. 12. Carga del archivo en la plataforma

Autor: Byron Delpino

- **Agregar Links**

Permite la observación y acceso de archivos, videos, imágenes disponibles en otros sitios web.

Clic en **Activar Edición**, **Añadir una Actividad o Recurso, URL, Agregar.**

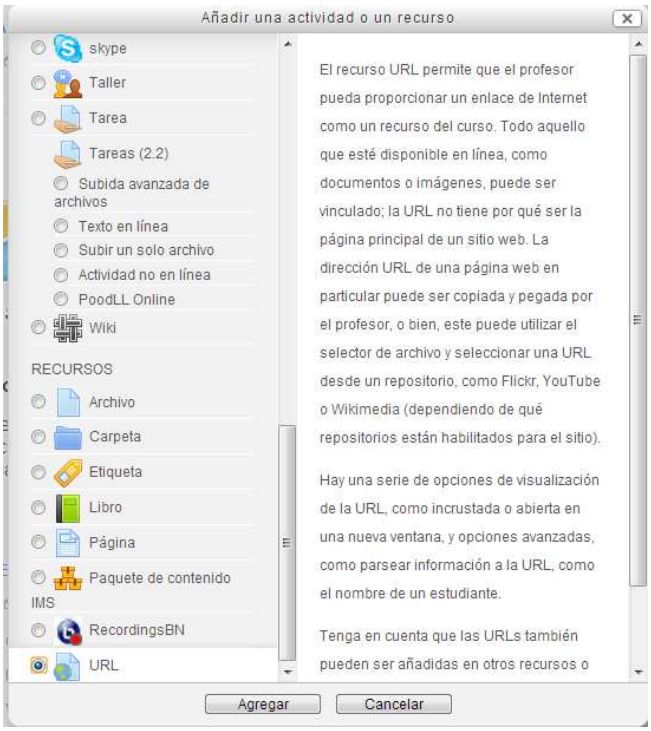


Figura 2. 13. Agregar Links en la Plataforma

Autor: Byron Delpino

Se agrega el nombre y descripción del recurso, así como la URL o sitio donde está disponible. Clic en **Guardar Cambios y Regresar al curso**.

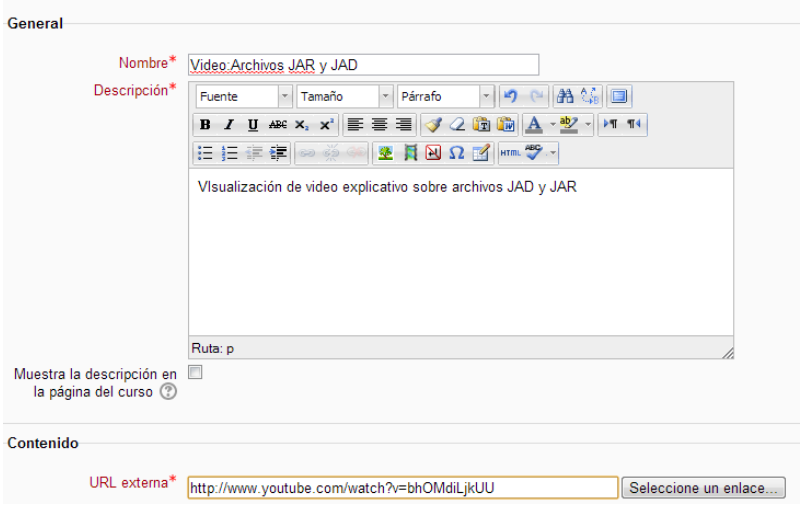


Figura 2. 14. Edición de nombre, descripción y sitio URL

Autor: Byron Delpino

HERRAMIENTAS DE COMUNICACIÓN Y EVALUACIÓN

Comunicación

Moodle facilita la comunicación entre la comunidad educativa mediante chats, encuestas, foros, wikis.

- **Agregar foros**

Clic en **Activar Edición**, **Añadir una Actividad o Recurso, Foro, Agregar.**

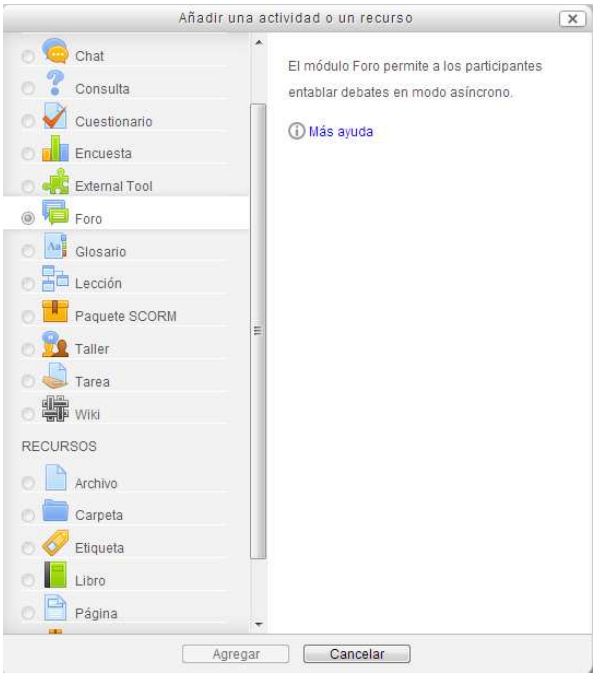


Figura 2. 15. Agregar foros en la plataforma
Autor: Byron Delpino

Se establece parámetros como nombre e introducción del foro, tamaño máximo y número de archivos adjuntos. Clic en **Guardar Cambios y Regresar al curso**.

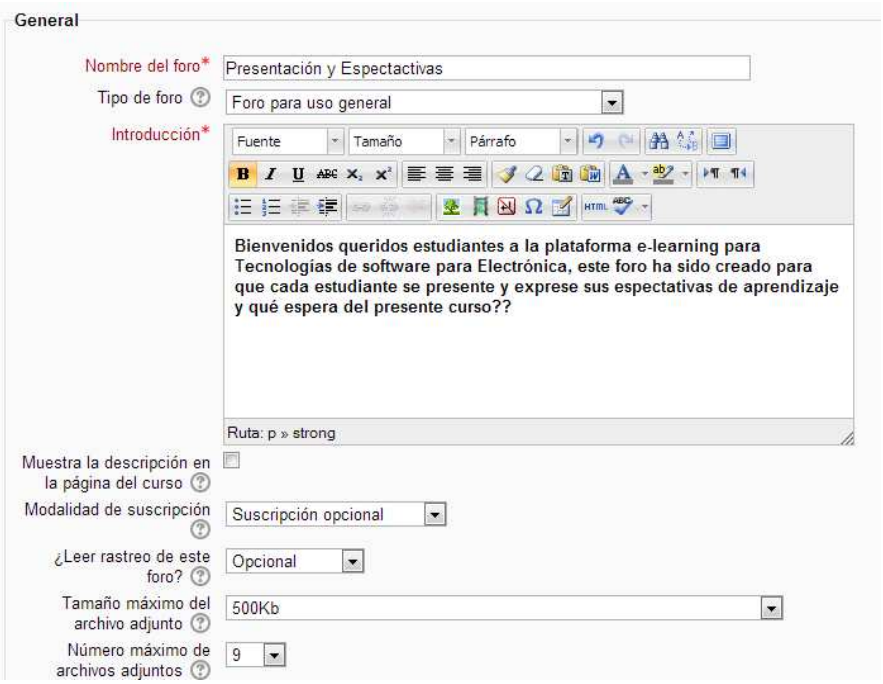


Figura 2. 16. Configuración de parámetros para agregar foros

Autor: Byron Delpino

Evaluación

La plataforma permite la evaluación de un estudiante mediante cuestionarios, tareas.

- **Agregar Cuestionario**

Clic en **Activar Edición**, **Añadir una Actividad o Recurso, Cuestionario, Agregar.**

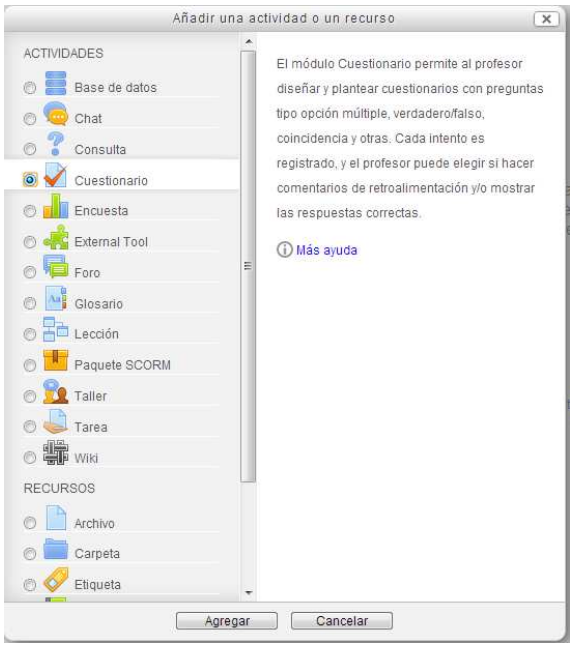


Figura 2. 17. Agregar Cuestionario en la plataforma

Autor: Byron Delpino

Se configura parámetros como el nombre, introducción, límite de tiempo, número de intentos permitidos. Clic en **Guardar Cambios y Regresar al curso**.

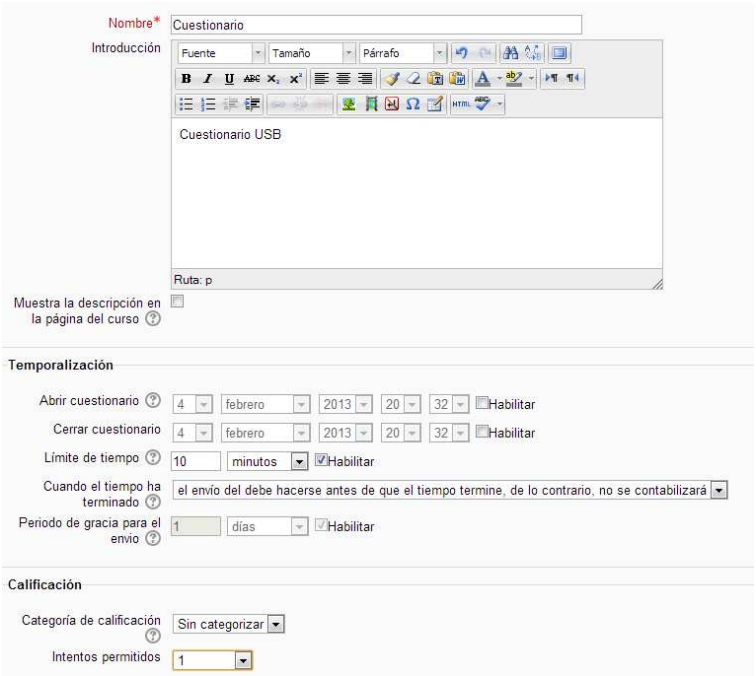


Figura 2. 18. Configuración de un cuestionario en la plataforma

Autor: Byron Delpino

Sobre a ventana de contenidos clic sobre el cuestionario creado, a continuación se agrega preguntas. **Editar Cuestionario**.

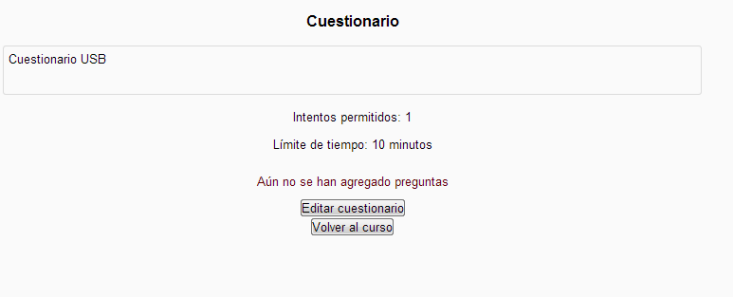


Figura 2. 19. Edición de Cuestionario en Moodle

Autor: Byron Delpino

Al presionar **Agregar una pregunta,** se despliega una ventana con un varios tipos de preguntas (Calculadas, opción múltiple, respuesta corta).

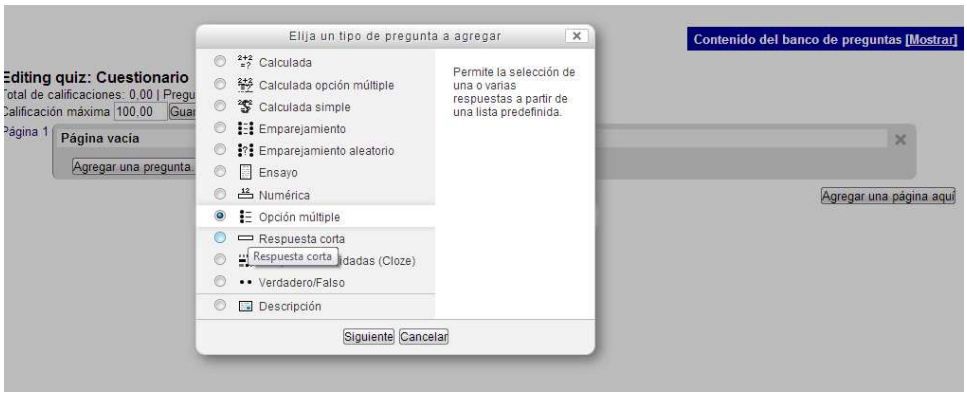


Figura 2. 20. Selección de tipo de pregunta a insertar en cuestionario

Autor: Byron Delpino

- **Opción Múltiple**

La Opción múltiple permite insertar preguntas con una o varias respuestas. Se edita la pregunta y se inserta la opciones a elegir y su respectiva calificación (Ejm 33% si son 3 respuestas).

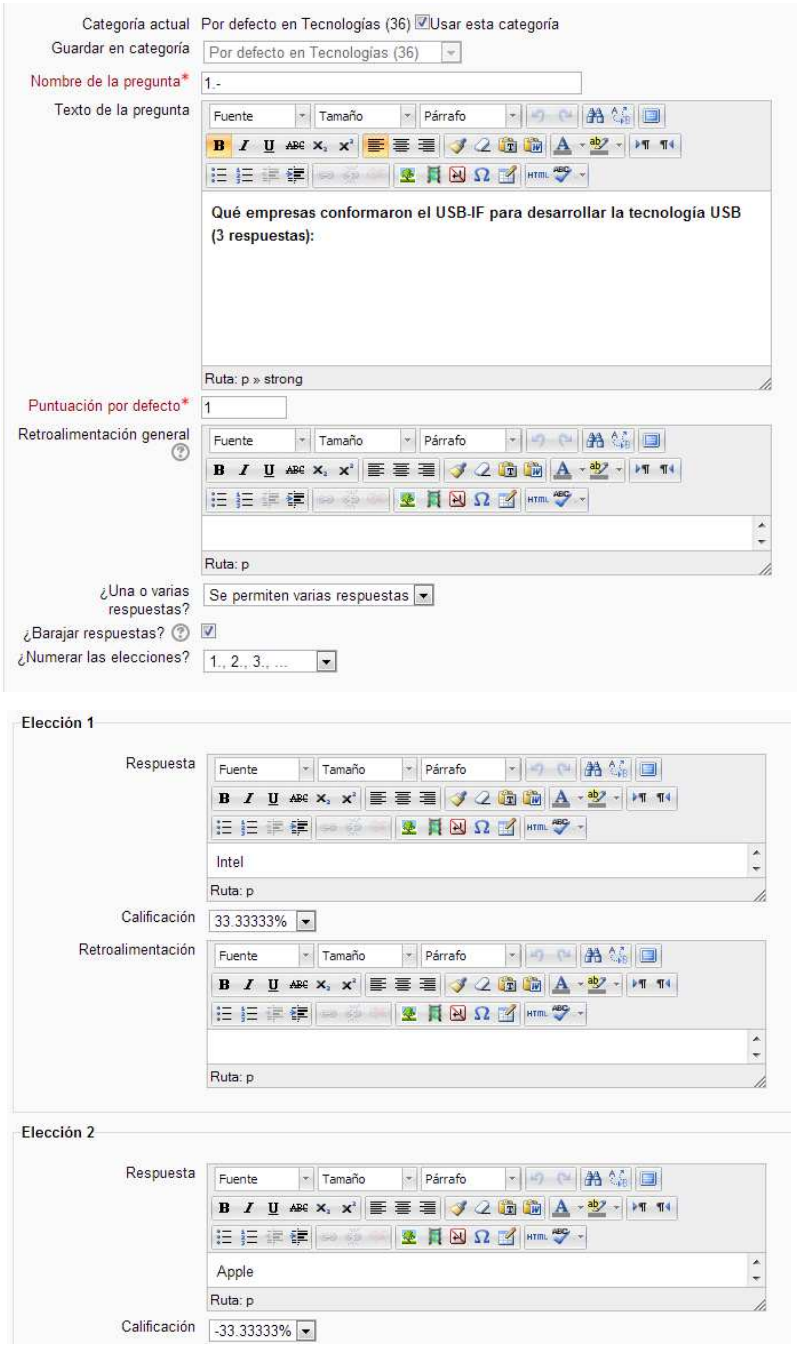


Figura 2. 21. Edición de una pregunta de opción múltiple

Autor: Byron Delpino

- **Preguntas tipo Cloze**

Permite la inserción de preguntas para completar espacios, opción múltiple a través de códigos. Ejm {1: SHORTANSWER=2.0}.

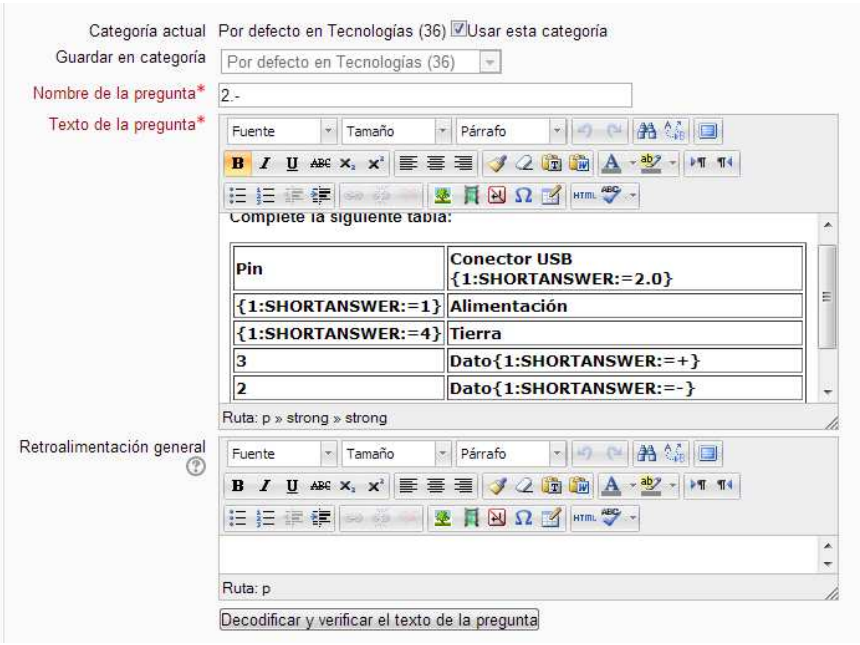


Figura 2. 22. Edición de una pregunta tipo cloze

Autor: Byron Delpino

HERRAMIENTAS DE ADMINISTRACIÓN Y ASIGNACIÓN DE PERMISOS

Moodle cuenta con herramientas para la administración del sitio, asignación de roles y permisos para cada uno de los usuarios de la plataforma.

- **Agregar y Matricular Usuarios**

Para matricular un grupo de usuarios al aula virtual de forma manual se requiere de un fichero codificado en utf-8 con los campos: nombre de usuario, nombre, apellido, contraseña, correo electrónico.

Ejm:

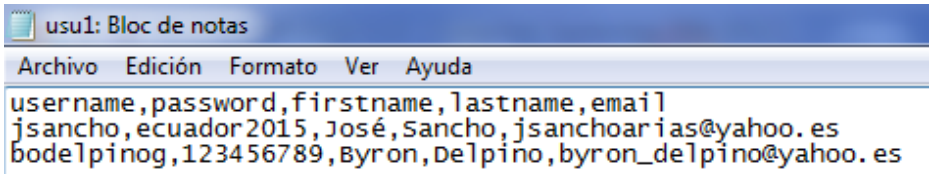


Figura 2. 23. Archivo .txt con los usuarios a matricular

Autor: Byron Delpino

Clic en **Activar Edición, Administración del Sitio, Usuarios, Cuentas, Subir Usuarios.**

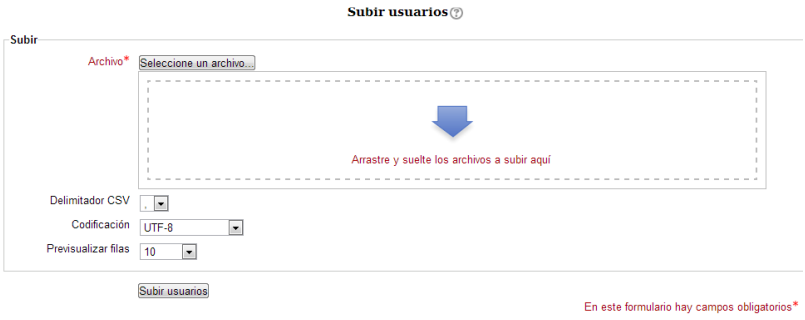


Figura 2. 24. Subir usuario a través de un archivo .txt

Autor: Byron Delpino

Se determina la ruta del archivo .txt, **Subir Archivo**. A continuación se despliega una ventana con los usuarios a subir en la plataforma. **Subir Usuarios, Continuar.**

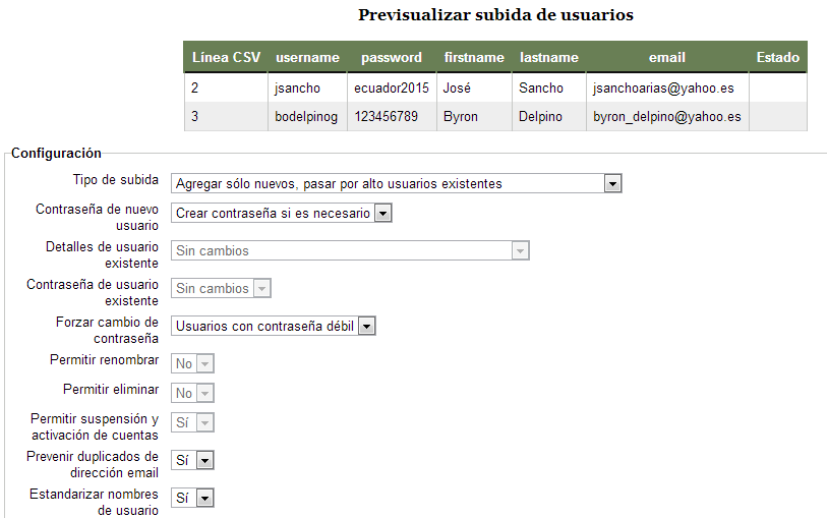


Figura 2. 25. Previsualización de usuarios a subir en la plataforma

Autor: Byron Delpino

Para la matriculación de los usuarios, clic en **Administración del Sitio, Usuarios, Usuarios Matriculados, Matricular Usuarios.**

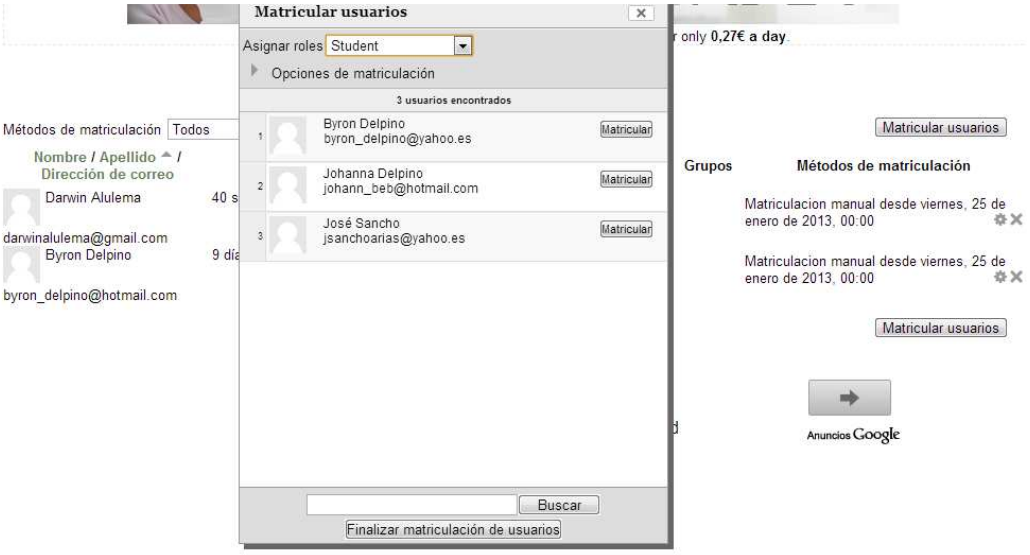


Figura 2. 26. Matriculación de Usuarios

Autor: Byron Delpino

Se determina los roles del usuario y se matricula.

Figura 2. 27. Lista de Usuarios Matriculados

Autor: Byron Delpino

CAPÍTULO 3

PRUEBAS Y AJUSTES DE LA PLATAFORMA

La plataforma virtual fue implementada en los sitios:

- http://doalulema.gnomio.com/
  Usuario Administrador: admin

- http://ueqs.org/virtual/
  Usuario Administrador: virtualadmin

3.1. SESIÓN COMO ESTUDIANTE

Visualización del aula virtual: Página de Inicio

Figura 3. 1. Página de Inicio de la Plataforma

Autor: Byron Delpino

Acceso a Archivos

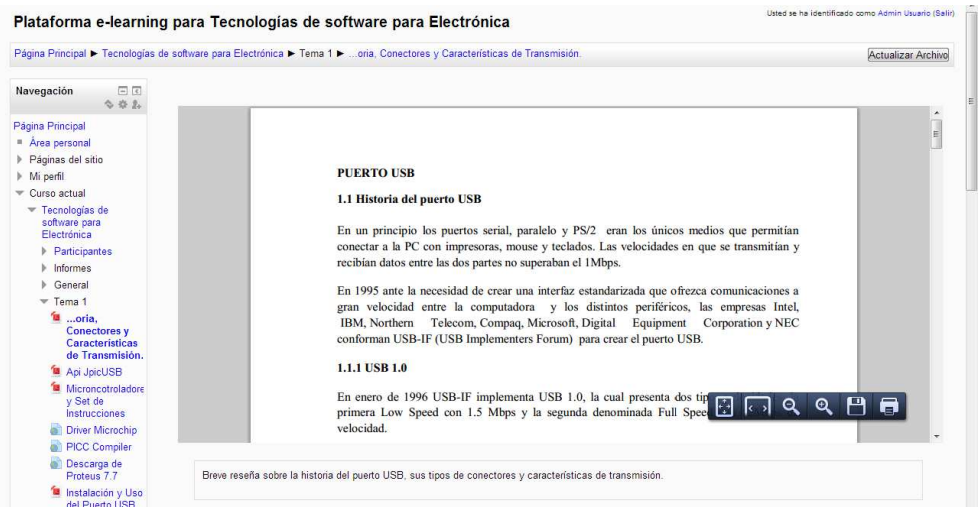


Figura 3. 2. Visualización de archivos cargados en la plataforma

Autor: Byron Delpino

Acceso a Vínculos Externos

Figura 3. 3. Acceso a Videos disponibles en sitios web externos

Autor: Byron Delpino

Evaluaciones

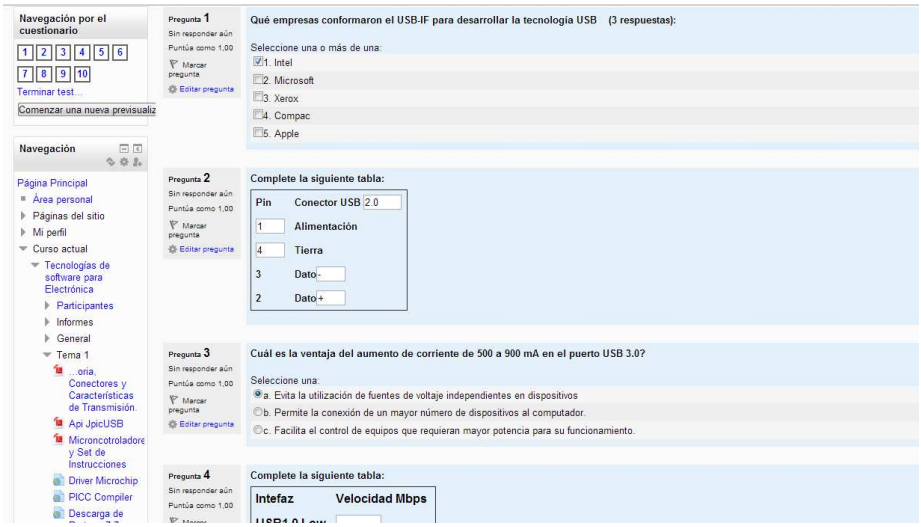


Figura 3. 4. Evaluación ejecutada por el usuario

Autor: Byron Delpino

Participaciones en Foros

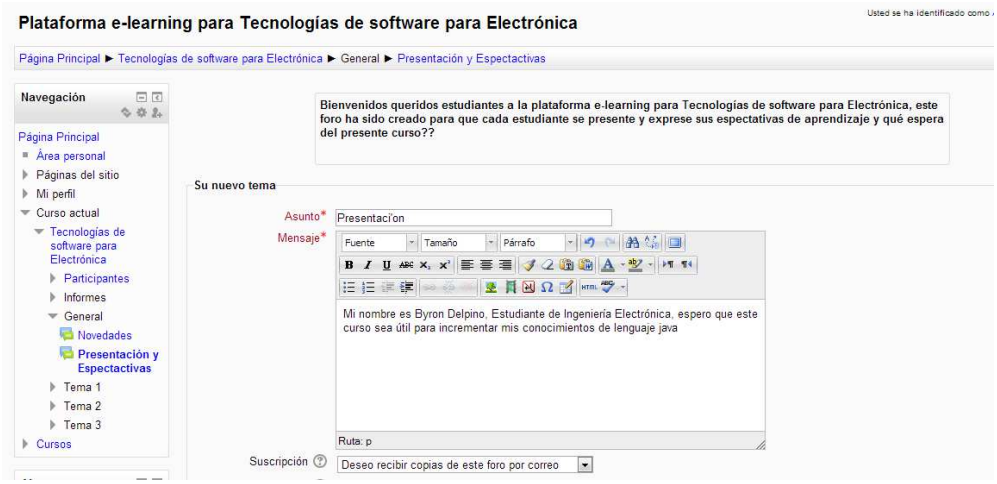
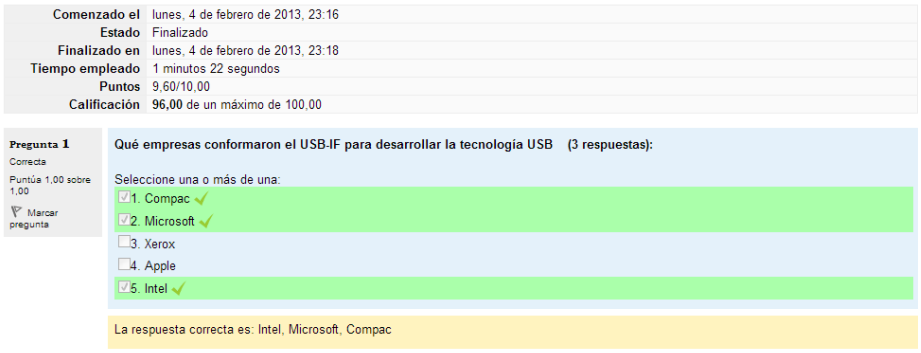


Figura 3. 5. Participación del estudiante en foros

Autor: Byron Delpino

Calificaciones



Comenzado el	lunes, 4 de febrero de 2013, 23:16
Estado	Finalizado
Finalizado en	lunes, 4 de febrero de 2013, 23:18
Tiempo empleado	1 minutos 22 segundos
Puntos	9,60/10,00
Calificación	**96,00** de un máximo de 100,00

Pregunta 1
Correcta
Puntúa 1,00 sobre 1,00
Marcar pregunta

Qué empresas conformaron el USB-IF para desarrollar la tecnología USB (3 respuestas):

Seleccione una o más de una:
- 1. Compac
- 2. Microsoft
- 3. Xerox
- 4. Apple
- 5. Intel

La respuesta correcta es: Intel, Microsoft, Compac

a) Examen Corregido

Usuario - Byron Delpino

Ítem de calificación	Calificación	Rango	Retroalimentación
Tecnologías de Software para Electrónica			
Cuestionario Puerto USB	96,00	0–100	
Cuestionario JSF	-	0–100	
Cuestionario J2ME	-	0–100	
Total del curso	***96,00***	***0–100***	

b) Calificaciones

Figura 3. 6. Visualización de evaluaciones y calificaciones del estudiante

Autor: Byron Delpino

Rol del Estudiante

- ✓ Acceder a los contenidos: Archivos, videos, sitios de descargas.
- ✓ Participar en foros, comunicarse con el docente u otros compañeros.
- ✓ Realizar evaluaciones.
- ✓ Visualizar la corrección de sus evaluaciones una vez que estas hayan terminado.
- ✓ Cumplir con actividades de evaluación a través de elaboración de Productos de la Unidad.
- ✓ Obtener un reporte de sus calificaciones.

3.2. SESIÓN COMO INSTRUCTOR

Visualización del aula virtual: Página de Inicio

Figura 3. 7. Página de Inicio de la Plataforma

Autor: Byron Delpino

Visualización del aula virtual: Página de Inicio/ Activar Edición

Figura 3. 8. Página de Inicio de la Plataforma/ Activar Edición

Autor: Byron Delpino

Participaciones en Foros

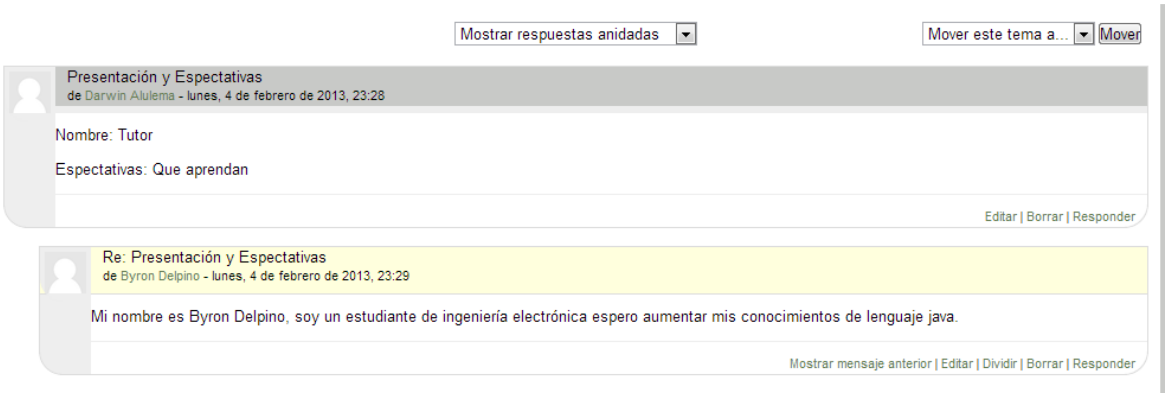


Figura 3. 9. Participaciones en Foros

Autor: Byron Delpino

Seguimiento de Actividades del Estudiantes

Calificador

Apellido / Nombre	Dirección de correo	Tecnologías de Software ... Cuestionario Puerto USB	Cuestionario JSF	Cuestionario J2ME	Total del curso
Darwin Alulema	darwinalulema@gmail.com	-	-	-	-
Byron Delpino	byron_delpino@yahoo.es	96,00	-	-	96,00
Byron Delpino	byron_delpino@hotmail.com	100,00	-	-	100,00
Johanna Delpino	johann_beb@hotmail.com	-	-	-	-
Juan Perez	jperez@hotmail.com	57,50	-	-	57,50
Promedio general		84,50	-	-	84,50

Figura 3. 10. Calificaciones de los estudiantes

Autor: Byron Delpino

Rol del Instructor

- ✓ Creación y administración de cursos.
- ✓ Creación y edición de contenidos (archivos, videos, sitios de descargas).
- ✓ Gestión de matriculas.
- ✓ Implementación de actividades de evaluación (cuestionarios, Producto de la Unidad).
- ✓ Acceso a reportes del curso.
- ✓ Participación en foros, comunicación con estudiantes.

CAPÍTULO 4

PROPUESTAS METODOLÓGICAS Y CURRICULARES

4.1. PROPUESTA DE GUIA METODOLOGICA PARA EL DISEÑO E IMPLEMENTACIÓN DE UNA PLATAFORMA E-LEARNING PARA LA MATERIA DE TECNOLOGÍAS DE SOFTWARE PARA ELECTRÓNICA

4.1.1. INTRODUCCIÓN

El permanente adelanto tecnológico ha permitido el crecimiento de una sociedad que emplea el Internet y las telecomunicaciones en la salud, economía, hogar y educación.

En la última década las Tecnologías de la Información y Comunicaciones (TICs) se presentan como herramientas idóneas en el proceso educativo, donde el profesor y el estudiante incorporan una visión constructivista orientada a generar conocimiento, fomentando habilidades y competencias en el alumnado.

E-Learning es un método de educación a distancia completamente virtualizado a través del uso de las Tics, facilitando un aprendizaje interactivo, flexible y accesible, sin embargo la mayoría de cursos virtuales no plasman las metodologías para una correcta enseñanza que aproveche la tecnología.

E-learning se presenta como un instrumento relevante para la enseñanza de cualquier temática, más aun en cuanto a impartir conocimientos sobre programación JAVA, siendo un soporte y complemento para docentes y alumnos que la utilicen como un elemento de consulta e investigación.

Es por ello que se propone una guía metodológica para el Diseño de una Plataformas E-Learning para la materia de Tecnologías de Software para Electrónica

4.1.2. DESARROLLO DE UNA ESTRATEGIA METODOLÓGICA

El diseño e implementación de una plataforma e-learning consiste en el desarrollo de contenidos que se adapten a los medios tecnológicos (recursos multimedia, Internet) y cumplan con los objetivos educativos.

A continuación se describen brevemente los pasos que se deberán seguir en el diseño y construcción de un curso de aprendizaje virtual.

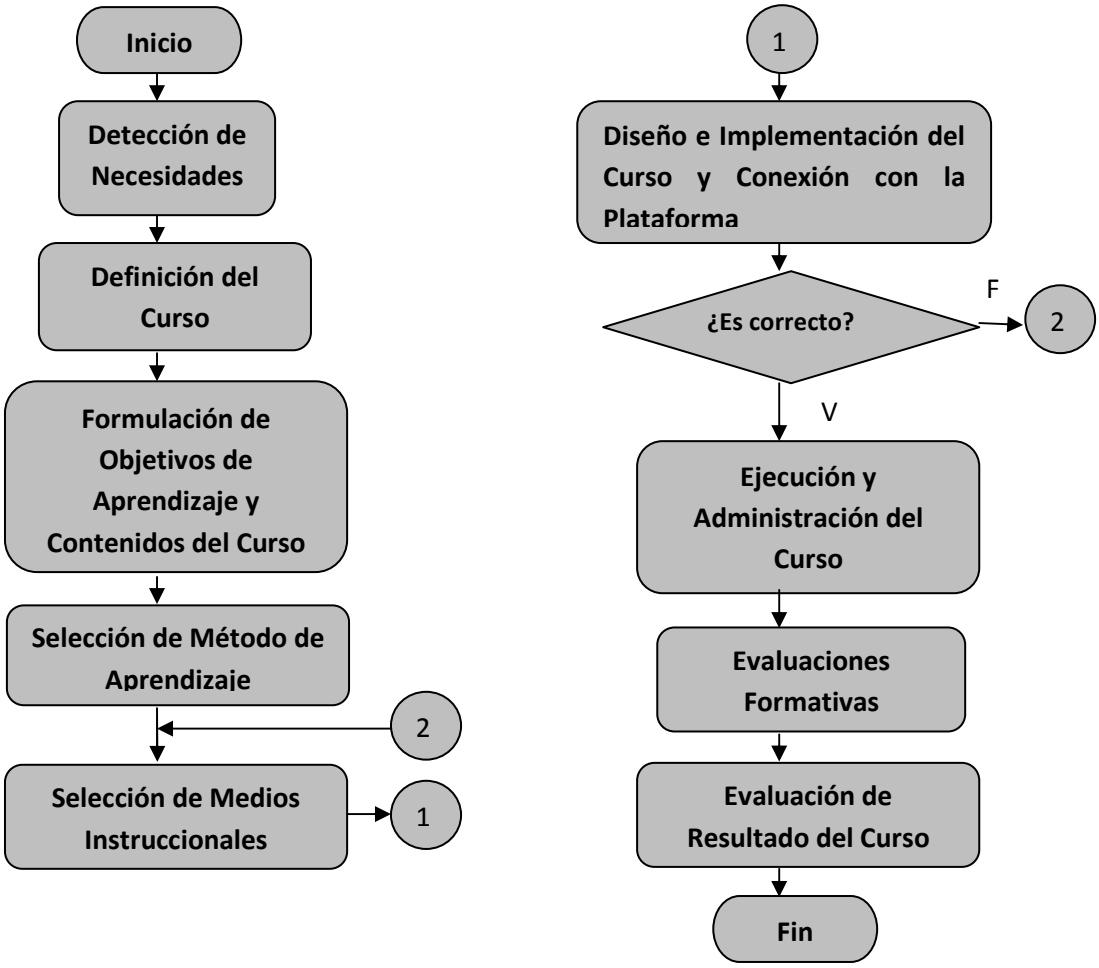


Figura 4. 1. Diagrama de Desarrollo de la Plataforma Virtual

Autor: Byron Delpino

4.1.2.1 Detección de Necesidades

Orientada a descubrir las necesidades y exigencias planteadas por los actores de la comunidad educativa. La detección de necesidades se da mediante encuestas, cuestionarios, experiencia del docente con respecto al desempeño de sus alumnos.

4.1.2.2 Definición del Curso

Consiste en la participación de expertos en tecnología y desarrollo educacional para el diseño e implementación de la plataforma. Dentro de la definición se debe incluir el nombre del curso, así como una descripción global del sus contenidos.

4.1.2.3 Formulación de Objetivos de Aprendizaje y Contenidos del Curso

Abarca las competencias a desarrollar en el estudiante.

La estructuración de contenidos se orientan a:

- Cubrir los objetivos del curso.
- Crear conocimiento útil.
- Desarrollo de actitudes y capacidades.

2.4 Selección de Métodos de Aprendizaje

Para la creación de material didáctico se emplea los métodos conductistas, cognitivo o constructivista, así como el método PACIE para la implementación de cursos virtuales.

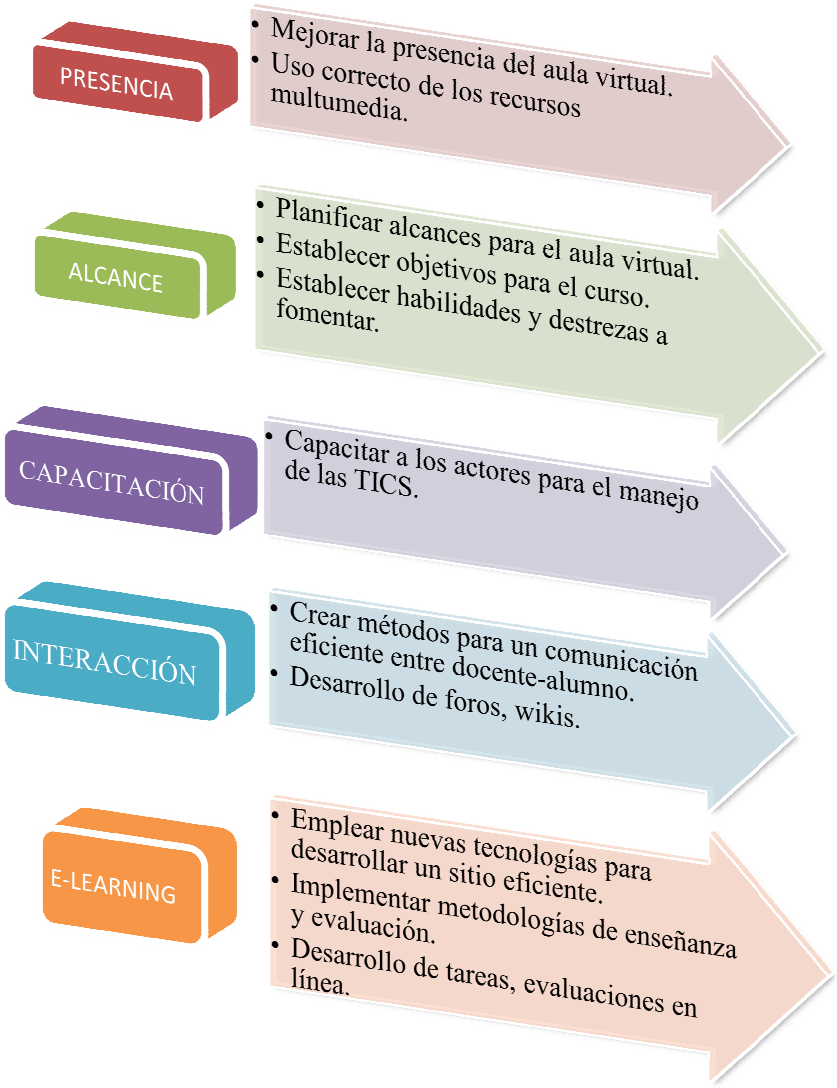


Figura 4. 2. Procesos de la metodología PACIE

Autor: Byron Delpino

4.1.2.5 Selección de Medios Instruccionales

Los medios instruccionales son aquellos que proporcionan información, guían los aprendizajes, ejercitan habilidades, motivan, y evalúan a través de las modalidades asíncrona (documento, videos, animaciones) y síncrona (foros, chats, wikis).

4.1.2.6 Diseño e Implementación del Curso y Conexión con la Plataforma

Consiste en la implementación de módulos e interfaces de la plataforma, además de los recursos a emplear para el procesamiento y almacenamiento de la información.

4.1.2.7 Ejecución y Administración del Curso.

La elaboración de un curso consiste en su implementación en una plataforma LMS (Moodle) y en su administración (corregir errores, desarrollar contenido extra para motivar al estudiante, evaluación del desempeño de la plataforma), de esta forma se busca un producto que cumpla con los objetivos de aprendizaje.

4.1.2.8 Evaluaciones Formativas

La evaluación cumple un rol determinante en la formación de un estudiante, es por ello que la plataforma e-learning debe contar con métodos de evaluación integrales y sistemáticos.

Una evaluación se la puede aplicar:

- A comienzo del curso (prueba de diagnóstico).
- Al finalizar un tema o unidad.
- Al terminar el curso.

4.1.2.9 Evaluación del Resultado del Curso

Orientado a evaluar si los objetivos de aprendizaje se cumplieron. Esta evaluación se basa en el número de estudiantes que aprobaron o concluyeron con éxito el curso. La Evaluación del Resultado del curso busca evaluar al docente y a la forma como desarrolló el curso.

4.2. PROPUESTA DE PLAN MICROCURRICULAR

4.2.1. DATOS INFORMATIVOS

COMPETENCIAS COMUNES DE LA CARRERA DE INGENIERÍA ELECTRÓNICA

A. Resuelve problemas relacionados a la ingeniería electrónica con iniciativa, aplicando sólidos conocimientos físico, matemáticos e instrumentales necesarios para interpretar y valorar la aplicación de nuevos conceptos y desarrollos tecnológicos.
B. Ejecuta proyectos en el ámbito de la electrónica con responsabilidad, de acuerdo a estándares de procedimientos internacionales.
C. Aplica técnicas de programación e implementa dispositivos electrónicos de última tecnología para disminuir la dependencia tecnológica del país, cumpliendo normas internacionales para la documentación y la elaboración de sus diseños.

UNIDADES DE COMPETENCIA EN RELACIÓN A LAS COMPETENCIAS

A.1. Entiende, relaciona y conceptualiza los métodos y teorías matemáticos.
A.2. Analiza y evalúa el procesamiento y modelamiento matemático de señales y sistemas.
A.3. Estudia y analiza el comportamiento de los fenómenos físicos en los dispositivos semiconductores y campos electromagnéticos.

B.1. Adquiere dominio en el manejo y utilización eficiente de los equipos de generación y medida vinculados con el desarrollo de proyectos de la ingeniería electrónica.
B.2. Establece procedimientos experimentales de baja y alta potencia, baja frecuencia; combinando instrumentos de generación y medida, así como los fundamentos de los circuitos eléctricos y electrónicos.

C.1. Analiza el problema, desarrolla la lógica de programación e implementa el software específico para la solución del mismo, así como el análisis y desarrollo de las redes básicas de computadoras y sus servicios y aplicaciones.
C.2. Analiza y desarrolla hardware electrónico utilizando circuitos digitales de baja, mediana y muy alta escala de integración.

ELEMENTO DE COMPETENCIA (UNIDAD DE COMPETENCIA C.1. - C.2.)

Analiza el problema, desarrolla la lógica de programación e implementa el software específico para la solución del mismo, así como el análisis y desarrollo de las redes básicas de computadoras y sus servicios y aplicaciones.

<table>
<tr><td>ASIGNATURA:

TECNOLOGIAS DE SOFTWARE PARA ELECTRONICA</td><td>CÓDIGO:

ELEE 24089</td><td colspan="2">NIVEL:

CUARTO</td><td>CRÉDITOS:

6</td></tr>
<tr><td>DEPARTAMENTO:

ELÉCTRICA Y ELECTRÓNICA</td><td colspan="2">CARRERAS:

INGENIERIA ELECTRONICA EN
• TELECOMUNICACIONES
• AUTOMATIZACION Y CONTROL
• REDES Y COMUNICACIÓN DE DATOS
• INSTRUMENTACION</td><td colspan="2">ÁREA DEL CONOCIMIENTO:

SISTEMAS DIGITALES</td></tr>
<tr><td rowspan="2">PERÍODO ACADÉMICO:

MARZO 2013 – JULIO 2013

FECHA ELABORACIÓN:

10/ENERO/2013</td><td colspan="2">SESIONES/SEMANA:</td><td rowspan="2">EJE DE FORMACIÓN:

PROFESIONAL</td></tr>
<tr><td>TEÓRICAS:

2 H</td><td>PRACTICAS:

4 H</td></tr>
<tr><td colspan="5">PRE-REQUISITOS:

CIRCUITOS DIGITALES (ELEE 14006)

PROYECTO INTEGRADOR I (ELEE 15063)

PROGRAMACION II (ELEE 15083)</td></tr>
<tr><td colspan="5"><u>DESCRIPCIÓN DE LA ASIGNATURA:</u>

Tecnologías de Software para Electrónica es una asignatura de formación intermedia del área de profesionalización, en la que se revisan los principios de control del puerto serial y usb, programación de páginas web, base de datos.

Esta asignatura pretende crear las competencias necesarias del futuro profesional para que realice</td></tr>
</table>

procesos de análisis, diseño e implementación de algoritmos de programación orientada a objetos para el control telemático de dispositivos electrónicos y almacenamiento de información utilizando patrones de diseño.

UNIDADES DE COMPETENCIAS A LOGRAR:

GENÉRICAS:

1. Demuestra en su accionar profesional valores universales y propios de la profesión, demostrando inteligencia emocional y creatividad en el desarrollo de las ciencias, las artes, el respeto a la diversidad cultural y equidad de género.
2. Interpreta y resuelve problemas de la realidad aplicando métodos de investigación, métodos propios de las ciencias administrativas, herramientas tecnológicas y diversas fuentes de información en idioma nacional y extranjero, con honestidad, responsabilidad, trabajo en equipo y respeto a la propiedad intelectual.

ESPECÍFICAS:

1. Resuelve problemas relacionados con la ingeniería electrónica con iniciativa, aplicando sólidos conocimientos matemáticos, científicos, tecnológicos e instrumentales produciendo soluciones acordes al desarrollo tecnológico.
2. Gestiona proyectos experimentales en el ámbito de la electrónica con responsabilidad, de acuerdo a estándares de procedimientos internacionales.
3. Aplica técnicas de programación y dispositivos electrónicos de última tecnología con creatividad, para disminuir la dependencia tecnológica del país con responsabilidad social, cumpliendo normas internacionales para la documentación y la elaboración de sus diseños.

ELEMENTO DE COMPETENCIA:

Analiza el problema, desarrolla la lógica de programación e implementa el software específico para la solución del mismo, así como el análisis y desarrollo de las redes básicas de computadoras y sus servicios y aplicaciones.

RESULTADO FINAL DEL APRENDIZAJE:

Software de control telemático de dispositivos electrónicos mediante el uso del lenguaje Java, C y motores de Bases de Datos, para resolver problemas reales.

CONTRIBUCIÓN DE LA ASIGNATURA A LA FORMACIÓN PROFESIONAL:

Esta asignatura corresponde a la etapa intermedia del eje de formación profesional, proporciona al futuro profesional las bases conceptuales para el diseño de aplicaciones de control y gestión telemáticas, con el apoyo de asignaturas del área de Sistemas Eléctricos, Electrónicos y Digitales

4.2.2. SISTEMA DE CONTENIDOS

No.	UNIDADES DE ESTUDIO Y SUS CONTENIDOS	EVIDENCIA DEL APRENDIZAJE Y SISTEMA DE TAREAS
1	**Unidad 1:** **CONTROL DE PUERTOS**	**Producto de unidad:** **SOFTWARE PARA EL CONTROL DE DISPOSITIVOS EXTERNOS DE BAJA Y MEDIA POTENCIA CONECTADOS A LA PC.**
	Contenidos de estudio: **1.1 INTERFACES DE LA PC** 1.1.1 Arquitectura de 32 y 64 bits 1.1.2 Tipos de Buses 1.1.3 Puerto Serie 1.1.3.1 Introducción 1.1.3.2 Conector DB9 1.1.3.3 Direcciones de puertos 1.1.3.4 RS-232 1.1.3.5 Drivers/Receivers 1.1.3.6 Comunicación Asíncrona 1.1.3.7 Comunicación Síncrona	**Tarea principal 1.1:** Resolución de problemas y ejercicios relacionados con el acondicionamiento de señales. **Tarea principal 1.2:** Práctica de laboratorio sobre: Manejo de puertos con Java

	1.1.4 Puerto USB 1.1.4.1 Historia del puerto USB 1.1.4.2 Conector USB 1.1.4.3 Características de Transmisión 1.1.4.4 Microcontroladores con puerto USB **1.2 ACONDICIONAMIENTO DE SEÑALES DIGITALES** 1.2.1 Tratamiento antirrebotes para pulsadores. 1.2.1.1 Circuitos para el tratamiento de antirrebote por hardware. 1.2.1.2 Algorítmica para el tratamiento de antirrebote por software. 1.2.2 Aplicación de circuitos electrónicos para protección y el aislamiento eléctrico de los dispositivos electrónicos del PC. 1.2.2.1 Relé electromecánico y de estado sólido 1.2.2.2 Optoacoplador 1.2.2.3 Triac 1.2.2.4 Conversor de niveles de voltaje. 1.2.2.5 Conversores ADC/DAC. **1.3 COMUNICACIONES CON LA PC** 1.3.1 API de Comunicaciones de Java 1.3.1.1 Configuración 1.3.1.2 Jerarquía de clases 1.3.2 Utilización de terminales de comunicaciones 1.3.3 Monitores de Puertos 1.1.4 API jPicUSB 1.1.4.1 Aplicaciones y Set de Instrucciones	

	del microcontrolador.	
2	**Unidad 2:** **APLICACIONES CLIENTE-SERVIDOR**	**<u>Producto de unidad:</u>** **APLICACION TELEMÁTICA (CLIENTE - SERVIDOR) PARA LA GESTIÓN DISPOSITIVOS ELECTRÓNICOS A TRAVÉS DE LA RED.**
	Contenidos de estudio: **2.1 CODIGO NATIVO JAVA** 2.1.1 Librerías de Enlace Dinámico y Estático 2.1.2 Tipos de datos nativos. 2.1.3 Acceso a los métodos y variables. 2.1.4 Desarrollo de una librería de enlace dinámico **2.2 INTRODUCCION A LAS REDES DE COMPUTADORAS** 2.2.1 Introducción 2.2.2 Modelo de referencia OSI y TCP/IP 2.2.3 Tipos de Redes 2.2.4 Direccionamiento IP y DHCP 2.2.5 Puertos Lógicos 2.2.6 Arquitectura cliente servidor **2.3 PROGRAMACION DE COMUNICACIONES EN RED** 2.3.1 TCP 2.3.1.1 Conexión mediante Sockets	**Tarea principal 2.1:** Desarrollo de software relacionado con código nativo. **Tarea principal 2.2:** Taller en laboratorio para la implementación de red punto a punto **Tarea principal 2.3:** Desarrollo de software relacionado con almacenamiento local de datos de dispositivos electrónicos.

	2.3.1.2 Establecimiento de comunicación cliente servidor 2.3.1.3 Transmisión de datos 2.3.1.4 Comunicación bidireccional 2.3.1.5 Aplicaciones telemáticas con Sockets 2.3.2 UDP 2.3.2.1 Conexión mediante Datagramas 2.3.2.2 Establecimiento de comunicación cliente servidor 2.3.2.3 Transmisión de datos 2.3.2.4 Comunicación bidireccional 2.3.2.5 Aplicaciones telemáticas con Datagramas **2.3 BASES DE DATOS** 2.3.1 Herramientas gráficas para la creación y manipulación de bases de datos. 2.3.2 Definición: bases de datos, tablas y registros. 2.3.3 Diseño de Base de datos 2.3.4 El modelo entidad relación 2.3.5 Creación de bases de datos 2.3.6 Manipulación de tablas 2.3.7 Relacionar tablas: uno a muchos, muchos a muchos. 2.3.8 Sentencias SQL 2.3.9 Almacenamiento local de datos de dispositivos electrónicos.	

	Unidad 3: **APLICACIONES DISTRIBUIDAS**	**<u>Producto de unidad:</u>** **APLICACION DISTRIBUIDA PARA LA GESTIÓN DE DISPOSITIVOS ELECTRÓNICOS CON PERSISTENCIA.**
3	**Contenidos de estudio:** **3.1 JAVA SERVER FACES** 3.1.1 Conceptos Generales y Fundamentación 3.1.2 Modelo Vista Controlador 3.1.3 Ciclo de Vida JSF 3.1.4 Estructura Básica de una Aplicación JSF 3.1.5 Creación y Uso de Managed Beans 3.1.6 Etiquetas JSF 3.1.7 Convertidores y Validadores 3.1.8 Integración con PrimeFaces **3.2 J2ME** 3.2.1 Introducción J2ME 3.2.2 Arquitectura J2ME 3.2.2.1 Maquinas Virtuales 3.2.2.2 Configuraciones 3.2.2.3 Perfiles 3.2.2.4 Paquetes Opcionales 3.2.3 Midlets 3.2.4 Interfaces Gráficas de Usuario 3.2.5 Comunicación HTTP	**Tarea principal 3.1:** Construir una interfaz JSF aplicando estándares de codificación en el manejo de componentes gráficos y con diferentes Frameworks (PrimeFaces u otros). **Tarea principal 3.2:** Desarrollo de aplicaciones JSF, integración de puertos y bases de datos. **Tarea principal 3.3:** Desarrollo de aplicaciones J2ME, integración JSF y manejo de puertos.

4.2.3. RESULTADOS Y CONTRIBUCIONES A LAS COMPETENCIAS PROFESIONALES

LOGRO O RESULTADOS DE APRENDIZAJE	NIVELES DE LOGRO			El estudiante debe
	A Alta	B Media	C Baja	
A. Aplicar Conocimientos en matemáticas, ciencia e ingeniería.		X		Utiliza fundamentos matemáticos para el desarrollo de algoritmos.
B. Diseñar, conducir experimentos, analizar e interpretar datos.	X			Diseña e implementa algoritmos y circuitos orientados a solucionar problemas de ingeniería electrónica.
C. Diseñar sistemas, componentes o procesos bajo restricciones realistas.	X			Diseña, dimensiona e implementa algoritmos y circuitos para el control telemático de dispositivos electrónicos.
D. Trabajar como un equipo multidisciplinario.			X	Realiza grupos de trabajo con compañeros de diferente especialidad a la suya.
E. Identificar, formular y resolver problemas de ingeniería.	X			Resuelve problemas prácticos de ingeniería electrónica, aplicando programación.
F. Comprender la responsabilidad ética y profesional.	X			Propone trabajos inéditos y de actualidad.

G. Comunicarse efectivamente.	**X**			Presenta oralmente el resultado de proyectos y genera artículos científicos.
H. Entender el impacto de la ingeniería en el contexto medioambiental, económico y global.	**X**			Comprende el impacto de software libre y el proceso de globalización.
I. Comprometerse con el aprendizaje continuo.	**X**			Investiga nuevas tecnologías aplicables a la Ingeniería Electrónica.
J. Conocer temas contemporáneos.	**X**			Entiende que Java es aplicable y evoluciona con las nuevas tecnologías.
K. Usar técnicas, habilidades y herramientas prácticas para la ingeniería.	**X**			Emplea diseños modulares para integrar software y hardware escalable.

4.2.4. FORMAS Y PONDERACIÓN DE LA EVALUACIÓN

TÉCNICAS E INSTRUMENTOS	1er Parcial	2do Parcial	3er Parcial
Tareas	**4**	**4**	**4**
Laboratorios/Informes			
Pruebas			
Trabajo de Investigación	**4**	**4**	**4**
Evaluación Conjunta	**6**	**6**	**6**
Producto Integrador	**6**	**6**	**6**
Total:	**20**	**20**	**20**

4.2.5. PROYECCIÓN METODOLÓGICA Y ORGANIZATIVA PARA EL DESARROLLO DE LA ASIGNATURA

Se emplearán variados métodos de enseñanza para generar un aprendizaje continuo, para lo que se propone la siguiente estructura:

- Se diagnosticará conocimientos y habilidades adquiridas anteriormente, al inicio del periodo académico.
- Con la ayuda del diagnóstico se indagará lo que conoce el estudiante, como lo relaciona, que puede hacer con la ayuda de otros, qué puede hacer solo, qué ha logrado y qué le falta para alcanzar su aprendizaje significativo
- A través de preguntas y participación de los estudiantes el docente recuerda los requisitos previos de aprendizaje que permite al docente conocer la base a partir de la cual incorporará nuevos elementos de competencia, en caso de encontrar deficiencias enviará tareas para atender los problemas individuales.
- Se planteará interrogantes a los estudiantes para que den sus criterios y puedan asimilar la situación problemática
- Se iniciará con explicaciones orientadoras del contenido de estudio, donde el docente plantea los aspectos más significativos, los conceptos, principios y métodos esenciales; y propone la secuencia de trabajo en cada unidad de estudio.
- Se buscará que el aprendizaje se base en el análisis y solución de problemas; usando información en forma significativa; favoreciendo la retención; la comprensión; el uso o aplicación de la información, los conceptos, las ideas, los principios y las habilidades en la resolución de problemas de programación.
- Se buscará la resolución de casos para favorecer la realización de procesos de pensamiento complejo, tales como: análisis, razonamientos, argumentaciones, revisiones y profundización de diversos temas.
- Se realizarán prácticas de laboratorio para desarrollar las habilidades proyectadas en función de las competencias.
- Las actividades en clase se basarán en un aprendizaje basado en problemas, para usar la información significativa; favorecer la retención; la comprensión; y el uso o aplicación de la información, los conceptos, las ideas, los principios y

las habilidades; resolución de problemas reales.

- Se planteará la resolución de casos, para favorecer la realización de procesos de pensamiento complejo, tales como: análisis, razonamientos, argumentaciones, revisiones y profundización de diversos temas.
- La evaluación cumplirá con las tres fases: diagnóstica, formativa y sumativa, valorando el desarrollo del estudiante en cada tarea y en especial en las evidencias del aprendizaje de cada unidad.

El empleo de las TIC en los procesos de aprendizaje:

- Las herramientas de Software necesarias para el desarrollo de la materia son: NetBeans y Eclipse como IDEs de JAVA, PIC Compiler, Visual C++ y MySQL.
- Para distribución de materiales de estudio, así como para la recepción de tareas se utilizara correo electrónico, página web y el aula virtual.

4.2.6. DISTRIBUCIÓN DEL TIEMPO

TOTAL HORAS	CONFERENCIAS ORIENTADORAS DEL CONTENIDO	CLASES PRÁCTICAS (Talleres)	LABORATORIOS	CLASES EVALUACIÓN	Trabajo autónomo del estudiante
96	**32**	**26**	**32**	**6**	**96**

4.2.7. TEXTO GUÍA DE LA ASIGNATURA

TITULO	AUTOR	EDICIÓN	AÑO	IDIOMA	EDITORIAL
1. Comunicaciones y bases de datos con JAVA a través de ejemplos	Jesús Bobadilla	PRIMERA	2003	Español	Alfaomega

4.2.8. BIBLIOGRAFÍA RECOMENDADA

TITULO	AUTOR	EDICIÓN	AÑO	IDIOMA	EDITORIAL
2. Java 2: Curso de Programación	Ceballos Javier	SEGUNDA	2003	Español	Alfaomega
3. Java 2: Manual de usuario y tutorial	Froufe Agustín	QUINTA	2010	Español	Alfaomega
4. SQL y Java	Melton Jim Eisenberg		2002	Español	Alfaomega
5. Aprenda rápidamente a programar microcontroladores PIC	Carlos Reyes	TERCER	2010	Español	
6. Tutorial JSF	Sicuma	-	-	Español	Sicuma
7. Desarrollo de Aplicaciones Moviles Android y J2ME	Jorge Nolasco Valenzuela	PRIMERA	2012	Español	MACRO

4.2.9. LECTURAS PRINCIPALES QUE SE ORIENTAN REALIZAR

LIBROS – REVISTAS – SITIOS WEB	TEMÁTICA DE LA LECTURA	PÁGINAS Y OTROS DETALLES
http://es.wikipedia.org/wiki/Netbeans	Descripción de NetBeans	Todo el documento
http://netbeans.org/downloads/index.html	Descargar Software NetBeans	Java SE o Java EE

http://www.oracle.com/technetwork/java/javase/downloads/jdk-7u2-download-1377129.html	Descargar JDK, disponible para varios Sistemas Operativos (OS).	JDK para el OS
http://dev.mysql.com/downloads/	Software Base de Datos MySQL	Paquete My SQL Installer
http://java.sun.com/docs/books/jni/	Programar con JNI	GuiadeProgramación y Especificación
http://es.wikipedia.org/wiki/HTML	Lenguaje HTML	Todo el documento y ver Enlaces
http://www.proactiva-calidad.com/java/principal.html	Java J2EE y Patrones de Diseño	
Bibliotecas digitales que maneja la institución		
Sitio web de plataforma e-learning http://doalulema.gnomio.com/ http://ueqs.org/virtual/	USB, J2ME, JSF	Toda la Plataforma

INEAMIENTO CURRICULAR

PETENCIAS ESIONALES	UNIDADES DE COMPETENCIA	RESULTADOS DE APRENDIZAJE	CRITERIOS DE DES
lemas relacionados a la ónica con iniciativa, conocimientos físico, nstrumentales necesarios valorar la aplicación de desarrollos tecnológicos.	A.1. Entiende, relaciona y conceptualiza los métodos y teorías matemáticos.	A1.1 Entiende los fundamentos matemáticos generales de la ingeniería	Solucionario de problemas genera con verificación de respuestas
		A.1.2. Entiende los fundamentos matemáticos de la ingeniería electrónica.	Solucionario de problemas de la in con verificación de respuestas
		A.1.3. Relaciona los fundamentos matemáticos y herramientas numéricas en fenómenos eléctricos.	Algoritmos de solución de problem eléctricos con verificación de resp comparación con la realidad
		A.1.4. Aplica los fundamentos matemáticos para el análisis de señales en el dominio de la frecuencia	Memoria técnica de solución a pro utilizando métodos teóricos
	A.2. Analiza y evalúa el procesamiento y modelamiento matemático de señales y sistemas.	A.2.1. Aplica el cálculo diferencial e integral para el estudio y modelamiento de sistemas lineales e invariantes en el tiempo, determinísticos y no determinísticos.	Memoria técnica de solución a pro utilizando métodos teóricos
		A.2.2. Aplica métodos y algoritmos para el tratamiento digital de señales.	Memoria técnica teórica de soluci simulados utilizando métodos com
		A.2.3. Analiza diferentes esquemas de comunicación analógica y digital basados en modelos matemáticos.	Memoria técnica de las diversas té comunicación que muestre el proc comprobación práctica respectiva
	A.3. Estudia y analiza el comportamiento de los fenómenos físicos en los dispositivos semiconductores y campos electromagnéticos.	A.3.1. Analiza el comportamiento de los campos y ondas electromagnéticas	Solucionario de problemas teórico comportamiento de campos y onda con verificación de resultados Memoria técnica de la simulación comportamiento de los campos ele magnéticos variantes en el tiempo, resultados
		A.3.2. Analiza el comportamiento de la propagación de las ondas electromagnéticas en diferentes medios.	Solucionario de problemas teórico propagación de ondas y su compor diferentes medios con verificación Memoria técnica del diseño de acc arbitrarias a la línea de transmisión concentrados, para la mayor transf
		A.3.3. Analiza y diseña dispositivos de acoplamiento entre un medio de transmisión y el espacio libre	Memoria técnica del análisis y dis entre un línea de transmisión y una sistemas de VHF, UHF y microon transferencia de potencia. Diseño y construcción de una ante VHF y UHF, para la mayor transfe al espacio libre.
ctos en el ámbito de la onsabilidad, de acuerdo a de procedimientos nacionales.	B.1. Adquiere dominio en el manejo y utilización eficiente de los equipos de generación y medida vinculados con el desarrollo de proyectos de la ingeniería electrónica.	B.1.2. Conoce los fundamentos básicos de la física cuántica y el comportamiento microscópico de los cristales semiconductores.	Caracterizar el funcionamiento de semiconductores pn utilizando m herramientas como gráficos, tabla
		B.1.3. Opera equipos de medición y generación eléctricos y electrónicos mediante procedimientos que permiten cuantificar parámetros físicos.	Mide variables eléctricas con el m osciloscopio con seguridad perso Ejecuta procedimientos adecuado integridad de los equipos y las per
		B.1.4. Desarrolla proyectos que involucran el análisis y síntesis de circuitos con dispositivos semiconductores básicos en un ambiente experimental real.	Calcula circuitos electrónicos sin respecto a las respuestas obtenida Deduce las fórmulas de cálculo pa circuitos con diodos y transistores
			Diseña sistemas electrónicos de p analógico (filtros, amplificadores)

		ambiente experimental real, e interpreta lo obtenidos.
		B.2.5. Analiza los dispositivos electrónico energía, utilizando técnicas de análisis de y fuerza en un ambiente experimental real resultados obtenidos.
cas de programación e ositivos electrónicos de gía para disminuir la ecnológica del país, s internacionales para la y la elaboración de sus liseños.	C.1. Analiza el problema, desarrolla la lógica de programación e implementa el software específico para la solución del mismo, así como el análisis y desarrollo de las redes básicas de computadoras y sus servicios y aplicaciones.	C1.1. Desarrolla software de gestión y pr información para el control de dispositivo
		C1.2. Desarrolla Software de control telem dispositivos electrónicos mediante el uso y motores de Bases de Datos, para resolve
		C1.3. Traduce un problema que puede res lógica matemática a una función lógica.
		C1.4. Relaciona funciones lógicas con co las implementa mediante circuitos combin
		C1.5. Interpreta un problema de secuencia tiempo mediante diagramas de estados
		C1.6. Traduce un problema planteado en a un circuito secuencial utilizando circuito
	C.2. Analiza y desarrolla hardware electrónico utilizando circuitos digitales de baja, mediana y muy alta escala de integración.	C2.1. Construye sistemas de hardware y s microprocesadores, que conlleven a la sol reales
		C2.2. Desarrolla e implementa aplicacione

NIVELES DE ABSTRACCION DEL CONOCIMIENTO

La aprehensión del conocimiento puede tener diferentes niveles de profundidad y capacidad para realizar una abstracción conceptual relativa a la capacidad de dominio de las competencias, unidades de competencia, elementos de competencia y núcleos del conocimiento. Tanto en el saber conocer, saber hacer y saber ser dentro de los elementos cognitivos, cognoscitivos del conocimiento y su relación con las capacidades actitudinales.
En este entorno los niveles de aprendizaje se pueden esquematizar en los siguientes niveles básicos:

- **NIVEL FAMILIARIZACION DEL CONOCIMIENTO:** Una instancia de diagnóstico del tema a través de parámetros de referencia del conocimiento, donde existe la conceptualización, categorización temática y sus referencias y relaciones con otras temáticas correlacionadas.

- **NIVEL DE REPRODUCCION DEL CONOCIMIENTO:** Un nivel de dominio del conocimiento para poder identificar, desarrollar, redactar y diagramar contenidos, y en general reproducirlos a través de diferentes instrumentos de verificación.

- **NIVEL DE PRODUCCION DEL CONOCIMIENTO:** El conocimiento aplicado a productos que evidencian la apropiación del mismo a través de nuevos componentes que reflejan la aplicación en diferentes niveles (prototipos, esquemas, subsistemas, sistemas).

- **NIVEL DE CREACION DEL CONOCIMIENTO:** Mayor nivel de profundización del conocimiento que además de niveles de producción y aplicación, realiza contribuciones e innovaciones que sirven de referencia para nuevos ciclos del conocimiento.

CICLOS DEL APRENDIZAJE

El ciclo de aprendizaje sigue un orden secuencial para mejorar la calidad de educación:

- **Participación:** El docente socializa con los estudiantes el tema que se va a tratar a través de una evaluación previa de conocimientos.
- **Exploración:** El alumno recopila información relacionada con el tema que se planteó en clase.
- **Explicación:** Obtención de ideas, conceptos y nuevo vocabulario a partir de la información recopilada. El docente aclara dudas y planteamientos del estudiante.
- **Elaboración:** El profesor reta al estudiante a resolver algún problema a partir del conocimiento obtenido.

TIPOS DE EVALUACION

- **FINALIDAD**

  - **Sumativa:** Tiene como meta evaluar los resultados alcanzados en función del uso que se va a desarrollar. Este tipo de evaluación se realiza al finalizar un periodo académico o ciclo de trabajo y va de acuerdo a los objetivos de enseñanza planteados en un inicio.

  - **Formativa:** Evaluaciones que buscan la formación integral del estudiante. El docente sirve como guía para regular, corregir y orientar el proceso educativo. En la plataforma virtual los laboratorios desarrollados sirven para la formación del alumno.

  - **Diagnóstica:** Evaluación de capacidades del estudiante previo al comienzo de un nuevo aprendizaje. Al inicio de la plataforma virtual se dispone de un foro donde el estudiante expondrá lo que conoce de programación Java.

- **EXTENSIÓN**

  - **Integradora:** Evalúa todas las capacidades y temas tratados en clase. En la plataforma se cuenta con una actividad del estudiante denomina "Producto de la Unidad", el cual busca integrar todos los temas revisados.
  - **Parcial:** Focalizada en evaluar parte del aprendizaje o un tema en particular. En la plataforma se dispone de cuestionarios para cada tema planteado.

- **PRODUCCION**

  - **Cualitativa:** Evalúa la calidad del proceso o producto realizado y el nivel de aprovechamiento del estudiante. La valoración se da mediante el uso de adjetivos. Ejm: Bueno-Malo, Cumple-No cumple. El Producto de Unidad será un tipo de evaluación cualitativa ya que deberá evaluar la calidad de trabajo desarrollado.

  - **Cuantitativa:** Centrada en la cantidad de procesos y objetivos alcanzados. El tipo de valoración se da a través de un número o porcentaje. La plataforma virtual cuenta con cuestionarios que evalúan cuantitativamente el nivel de aprendizaje del alumno.

INSTRUMENTOS DE EVALUACION

MEDICIÓN DE INDICADORES APLICANDO MODELO ABET

Es un modelo específico para carreras de ingeniería, tiene 9 categorías y pide formar 11 competencias.

CATEGORÍAS ABET

A. Estudiantes
B. OE (Objetivos Educativos)
C. Resultados del programa –habilidad

D. Mejoras del sistema
E. Plan de estudio
F. Docentes
G. Instalaciones
H. Apoyo y recursos financieros
I. Criterios del programa

COMPETENCIAS ABET

1. Aplicar conocimientos de las matemáticas, ciencias e ingeniería.
2. Diseñar y conducir experimentos, así como el analizar e interpretar datos
3. Diseñar sistemas, componentes o procesos que satisfagan necesidades.
4. Trabajar en equipos multidisciplinarios.
5. Identificar, Formular y resolver problemas de Ingeniería.
6. Comprender su responsabilidad profesional y ética.
7. Comunicarse efectivamente.
8. Entender el impacto de la Ing. en la solución de problemas globales y sociales
9. Comprometerse con el aprendizaje a lo largo de toda la vida.
10. Conocer temas de actualidad.
11. Usar técnicas, estrategias y herramientas de la ingeniería moderna.

MEDICIÓN DE LOS LOGROS DE APRENDIZAJE A TRAVÉS DE VARIABLES E INDICADORES

LOGRO O RESULTADOS DE APRENDIZAJE	NIVELES DE LOGRO			EL ESTUDIANTE DEBE	VARIABLES	INDICADORES
	A Alta	B Media	C Baja			
Aplica conocimientos de desarrollo de aplicaciones de software		X		Tener conocimientos básicos verificables de resolución de programas, algoritmos, en las fases del proceso de desarrollo de aplicaciones de software.	Lenguajes de Programación Experticia desarrollo de software	Nivel de profundidad en programación. Nivel de lógica de programación. Nivel de parametrización y optimización. Cantidad de programas implementados. Tipos de aplicaciones desarrolladas. Estándares de codificación.
Elabora software funcional cumpliendo normas de programación establecidas. Presenta aplicaciones de software que manejen periféricos del computador para control de dispositivos de alta escala de integración.		X		Tener conocimientos de lenguajes de programación para el manejo de dispositivos electrónicos a través del computador.	Número de Lenguajes de Programación	

CAPÍTULO 5

CONCLUSIONES Y RECOMENDACIONES

5.1. CONCLUSIONES

- Las Tecnologías de la Información y Comunicación (Tics) son un grupo de herramientas que favorecen un aprendizaje óptimo e innovador, acogido a los estándares pedagógicos y a las actuales demandas de la comunidad educativa a través del manejo de información relevante, repositorios virtuales, herramientas web 2.0, web 3.0, plataformas virtuales confiables y certificadas.

- Moodle es un Sistema de Gestión de Aprendizaje gratuito con una gran comunidad de desarrolladores que brindan soporte y actualización a la plataforma, de esta forma se escatima recursos a la hora de diseñar e implementar un aula virtual.

- La Plataforma E-learning para la materia de Tecnologías de Software para Electrónica fue implementada en Moodle debido a que cuenta con recursos para la creación y edición de contenidos que garantizan el aprendizaje del lenguaje de programación Java.

- Para el proceso de aprendizaje en la formación profesional, se debe contar con distintas herramientas e indicadores de diagnóstico y evaluación, que apunten a una enseñanza significativa y verificable.

- La propuesta de guía metodológica para el diseño e implementación de una plataforma e-learning para la materia de tecnología de software para electrónica es una importante herramienta para la determinación de objetivos de aprendizaje, desarrollo de contenidos, creación y administración de cursos.

- El lenguaje de programación Java es útil para el aprendizaje de nuevas tecnologías, ya que implementa un sinnúmero de Apis y librerías que facilitan la creación de aplicaciones independientes o empresariales en cualquier tipo de dispositivo.

- La comunicación pc-microcontrolador a través de la librería JPicUsb es una excelente alternativa para la implementación de aplicaciones en el campo de la domótica, robótica e industria automotriz, ya que aprovecha las ventajas de velocidad y estandarización del puerto Usb y las funcionalidades del microcontrolador.

- JSF es la mejor opción para el desarrollo de aplicaciones cliente-servidor ya que cuenta con las bondades de java para integrar lenguaje html con base de datos, manejo de puertos, separando adecuadamente las capas de presentación, vista y datos mediante el Modelo-Vista-Controlador.

- Gracias a J2ME se tiene la posibilidad de crear y ejecutar aplicaciones móviles con grandes prestaciones para el usuario, debido al bajo consumo de memoria y procesamiento de la máquina virtual KVM.

5.2. RECOMENDACIONES

- Es necesario capacitar al docente y al estudiante en el manejo adecuado de las tics y nuevas tecnologías para mejorar el proceso de aprendizaje.

- Se recomienda que el Departamento de Eléctrica y Electrónica de un mayor empuje a la implementación de aulas virtuales, ya que es un excelente complemento de aprendizaje para los miembros de la comunidad educativa.

- Para la implementación de aulas virtuales se debe contar con profesionales en pedagogía, administración de sitios web e ingeniería electrónica, de tal forma se garantiza el correcto aprendizaje del estudiante.

- En base al contenido revisado en la plataforma e-learning se recomienda realizar investigaciones o proyectos de grados orientados a resolver los problemas de la ciudad o el país.
- Se recomienda el empleo de software libre para la implementación de aplicaciones relacionadas con la ingeniería electrónica (Java), así como el desarrollo de cursos virtuales que faciliten el aprendizaje (Moodle).

REFERENCIAS BIBLIOGRÁFICAS

[1] Universal Serial Bus. [En línea]. Disponible en: http://es.wikipedia.org /wiki/Universal_Serial_Bus [2012, 01 de octubre].

[2] Oñativia, G. (2009). jPicUSB: Clase Java para comunicación USB con PICs usando API de Microchip. [En línea]. Tucumán-Argentina: Universidad Nacional de Tucumán. Disponible en: http://www.edutecne.utn.edu.ar/microcontrol_congr /comunicaciones/JPICUS.PDF [2012, octubre].

[3] USB. [En línea]. Disponible en: http://es.scribd.com/doc/62467609/USB-CCS [2012, octubre].

[4] JavaServer Faces. [En línea]. Disponible en: http://es.wikipedia.org/wiki /JavaServer_Faces [2012, 04 de noviembre].

[5] Tutorial de JavaServer Faces. [En línea]. Sicuma. Disponible en: http://www.sicuma.uma.es/sicuma/Formacion/documentacion/JSF.pdf [2012, noviembre].

[6] Nolasco, J. (2012). Desarrollo de Aplicaciones Móviles Android y J2ME. Lima: MACRO.

[7] Galvez, S. y Ortega L. (2003). Java a Tope: J2ME. Málaga-España: Edición Electrónica.

[8] Quiroz Valencia, Arturo (2011). Inclusión de las Tecnologías de la Información y las Comunicación como eje transversal en el Bachillerato en Ciencias. Tesis de

Maestría en Sistemas Informáticos Educativos, Universidad Tecnológica Israel, Quito.

[9] Carrión Salinas, Alex (2011). Estudio de los Procesos de Mejoramiento del Aprendizaje y la Calidad Educativa, Mediante la Capacitación Docente en la Aplicación de las TICs en los Centros Educativos Interculturales Bilingües de la Zona de Guamote Comunidad de Santa Cruz de Usubug. Tesis de Maestría en Sistemas Informáticos Educativos, Universidad Tecnológica Israel, Quito.

[10] Gallardo, J., Pérez, R. y Cortez, M. (2002). Desarrollo de una Guía Metodológica para la construcción d e cursos de e-Learning en las Unidades de Educación a Distancia de la UCN. [En línea]. Antofagasta-Chile: Universidad Católica del Norte. Disponible en: http://lsm.dei.uc.pt/ribie/docfiles/txt2003121512587tci%2021.pdf [2013, 02 de febrero].

[11] Bustamante, P. (s.f.). Componentes de una plataforma e-learning. [En línea]. Disponible en: http://www.e-aula.cl/2011/06/componentes-de-una-plataforma-e-learning/ [2013, 03 de febrero].

[12] Alvarez, L. (s.f.). Sistemas de Gestión de Aprendizaje. [En línea]. Valdivia-Chile: Universidad Austral de Chile. Disponible en: http://www.gita.cl/files/3_Sistemas_de_Gestion_de_Aprendizaje_v21.pdf [2013, 03 de febrero].

[13] Carreño, D., Gajardo, V., Parra, C. y Vera, C. (s.f.). Teorías del Aprendizaje: Teorías cognitivivas. [En línea]. Disponible en: http://www.slideshare.net/teorias-del-aprendizaje-cognitivo [2013, 10 de febrero].

[14] Molina, D. (2008). ¿QUÉ ES LA WEB 1.0, 2.0 3.0?. [En línea]. Disponible en: http://dimamoa.blogspot.com/2008/05/qu-es-la-web-10-20-30.html [2013, 10 de febrero].

[15] EVALUACIÓN. [En línea]. Disponible en: http://www.oocities.org/es/baldomeroab/t/evaluacion.htm#tipos [2013, 15 de febrero].