Universidad Carlos III de Madrid

**PROYECTO FIN DE CARRERA**

# DESARROLLO DE UNA APLICACIÓN PARA LA ADQUISICIÓN Y ANÁLISIS DE LA SEÑAL DE PRESIÓN INTRACRANEAL PARA LA DETECCIÓN DE HIDROCEFALIA NORMOTENSA

INGENIERÍA TÉCNICA SISTEMAS DE TELECOMUNICACIÓN

Autor: Alexandra Riboul Ojeda

Tutor: Paula de Toledo Heras

Codirector: Alfonso Lagares

Leganés, Junio de 2011

# AGRADECIMIENTOS

Quiero agradecer a todos las personas que me han apoyado durante la realización de este proyecto, en especial a mi familia y mi novio.

También a mi tutora, Paula de Toledo, por abrirme las puertas del campo en el que quiero proyectar mi carrera profesional, la bioingeniería.

Por último agradecer a Alfonso Lagares, codirector del proyecto, la atención prestada en todo momento, facilitándome el trabajo en la planta de neurocirugía del hospital Doce de Octubre de Madrid.

# ÍNDICE

# 1. INTRODUCCIÓN

# INTRODUCCIÓN

El hospital 12 de Octubre de Madrid ha planteado a la universidad Carlos III de Madrid un problema con una de sus máquinas, que finalmente es propuesto como proyecto fin de carrera.

Nos encontramos por tanto ante un proyecto del área de bioingeniería.

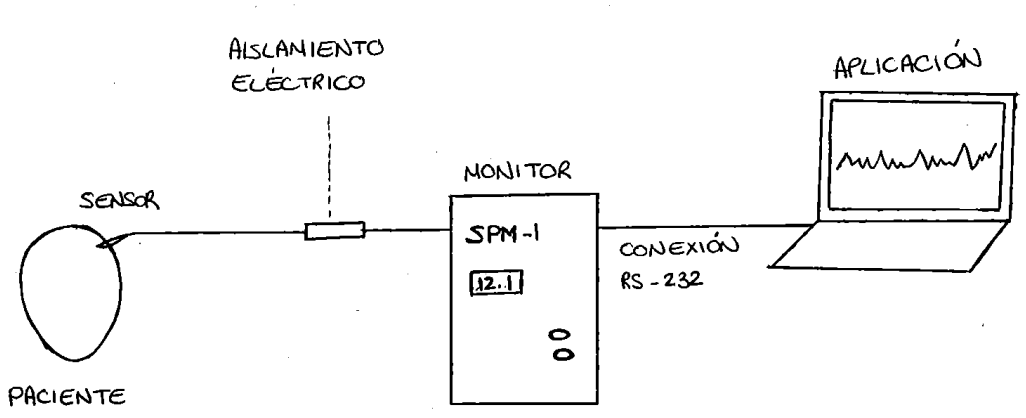
Este tipo de trabajos tienen la particularidad de necesitar de un estudio previo sobre el tema a tratar, que nos permitirá familiarizarnos con la situación y nos pondrá en disposición de poder dar una solución óptima y acorde al problema.

## 1.1 PRESENTACIÓN DEL PROBLEMA

La presión intracraneal (PIC) es aquella presión medida en el interior de la cavidad craneal. Un aumento significativo de ésta puede producir daños neurológicos permanentes, y en algunos casos puede ser mortal.  Para evitar estas situaciones se hacen monitorizaciones continuas de la presión intracraneal en los pacientes con un cuadro clínico tal, que los especialistas consideren oportuno llevar este control.

En la planta de neurocirugía del hospital 12 de Octubre cuentan con un equipo para monitorizar la presión intracraneal de los pacientes que así lo requieran.

El equipo consta de:




Como podemos ver, el equipo está formado por:

- Un sensor, que se introduce en el cráneo del paciente y recoge los valores de la PIC. Este sensor está eléctricamente aislado del equipo.
- Un monitor, que recoge los datos enviados por el sensor y los manda al ordenador.

- Un ordenador, que contiene una aplicación que recibe los datos enviados por el monitor. Esta aplicación va representando los datos recibidos en tiempo real, y así los especialistas pueden analizar las variaciones de la PIC con respecto al tiempo (curva de PIC).

Los profesionales de este campo consideran que la aplicación es bastante pobre, ya que no permite realizar las funciones necesarias para detectar ciertos patrones en una curva de PIC monitorizada, entorpeciendo así los diagnósticos.

## 1.2 OBJETIVO

El objetivo es realizar una aplicación, partiendo de cero, que cumpla todas las expectativas y requisitos de los especialistas, sencilla en su uso, y que facilite diagnosticar problemas con rapidez.

En concreto se trata del desarrollo de un software específico que permita el diagnóstico de hidrocefalia normotensa del adulto, el único tipo de demencia tratable y/o curable en la actualidad. El software debe permitir el análisis de la presión intracraneal de los pacientes, con lo que aunque la hidrocefalia sea en principio su primera utilidad, tendrá otras aplicaciones en otras patologías neurológicas y neuroquirúrgicas en el futuro.

Por último debe ser compatible con el equipo de medida del que disponen los especialistas, y adjuntar un manual de usuario, ya que finalmente esta aplicación es la que quedará en uso, desechando la antigua.

No está dentro del objetivo de este proyecto que la aplicación quede en uso inmediatamente después de su entrega, pues es necesario realizar diversas pruebas para asegurar su fiabilidad antes del uso en pacientes.

## 1.3 ESTRUCTURA DE LA MEMORIA

Esta memoria se ha estructurado en nueve apartados que se detallan a continuación.

- **Introducción.** Es el apartado actual en el que está recogido la presentación del problema y el objetivo del presente proyecto.

- **Análisis del problema.** En este apartado se obtienen los conocimientos necesarios para abordar el problema planteado.

- **Diseño y desarrollo de la aplicación.** Apartado en el que se describe el diseño de la aplicación y su consiguiente implementación.

- **Pruebas.** Apartado que describe las pruebas realizadas en la aplicación para comprobar su funcionamiento.

- **Resultados.** Apartado que recoge el resultado final obtenido.

- **Conclusiones y líneas futuras**. Apartado en el que se plantean las posibles líneas futuras de desarrollo del proyecto.

- **Presupuesto.** Apartado en el que se indican los costes estimados de la realización del proyecto.

- **Bibliografía.** Apartado que recoge las fuentes de documentación utilizadas en este proyecto.

- **Anexos.** Apartado en el que se exponen los anexos a los que ha dado lugar esta memoria.

# 2. ANÁLISIS DEL PROBLEMA

# INTRODUCCIÓN

Establecido el objetivo de este proyecto procedemos a obtener los conocimientos necesarios para abordar el problema planteado.

En primer lugar haremos una introducción a la presión intracraneal, al equipo de monitorización y estudiaremos las limitaciones de la aplicación actual en uso.

- 2.1 QUE ES LA PIC
- 2.2 COMO SE MIDE LA PIC
  - *2.2.1 Análisis de la aplicación anterior*

En segundo lugar se establecerán los requisitos de usuario para la nueva aplicación y se definirán los recursos disponibles para llevarla a cabo.

- 2.3 REQUISITOS
  - *2.3.1 Requisitos de usuario*
  - *2.3.2 Requisitos Software y Hardware*

Una vez documentados sobre los aspectos anteriores, estamos en posesión de poder tomar consecuentemente las decisiones de diseño acertadas.

## 2.1 QUE ES LA PIC

Se define como PIC, aquella presión medida en el interior de la cavidad craneal y que es el resultado de la interacción entre el continente (cráneo) y el contenido (encéfalo, líquido cefalorraquídeo y sangre).

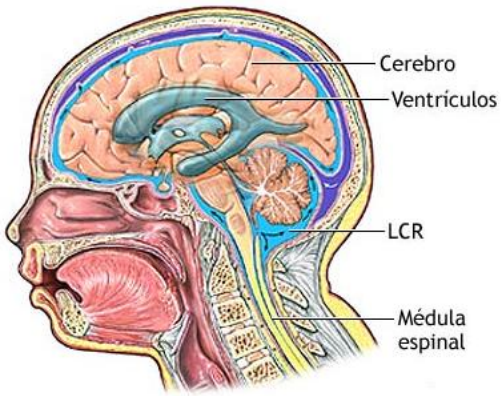
- **El continente** está formado por un recipiente aproximadamente esférico, el cráneo, constituido por una capa de hueso de variable grosor, además de poco o nada distensible en el adulto.

- **El contenido** de la cavidad craneana tiene tres componentes:

  - El encéfalo
    Es el tejido cerebral, normalmente es constante en volumen y ocupa el 80% del total.

  - Líquido cefalorraquídeo (10%) y sangre (10%).
    Estos dos elementos líquidos constituyen aproximadamente el 20% restante, dividido en proporciones iguales.

Estos componentes son poco compresibles por lo cual el aumento en uno de ellos debe compensarse con la disminución proporcional de los restantes, dicha ley es conocida como la Doctrina de Monro-Kellie.




*Doctrina Monro-Kellie*

El sumatorio de todos los volúmenes intracraneales es siempre una constante (k) mantenida por regulación interna.

Si ocurriera que se añade un volumen nuevo (por ejemplo, un tumor, un hematoma o una aneurisma) o se incrementara uno de los volúmenes ya existentes (por ejemplo, por un edema o hemorragia) los mecanismos compensatorios hacen que los otros volúmenes disminuyan proporcionalmente para mantener la constante (k) en el mismo valor.

En definitiva, si hay un aumento de volumen dentro del cráneo, tal, que no logra ser compensado, aumenta la presión intracraneal.

**PRESIÓN INTRACRANEAL NORMAL.**

Normalmente, la PIC fluctúa entre 1 y 15 mm Hg (mm Hg = milímetros de mercurio, es la unidad de medida)**,** no tiene un valor estable y se ve modificada por diversas situaciones fisiológicas (pulso, respiración, posición del individuo, tensión arterial, dolor) que cambian el volumen de los elementos del contenido.

Funciones fisiológicas responsables de la fluctuación de la PIC en un individuo sano:

1. El pulso cardiaco provoca una verdadera inyección de sangre dentro de los vasos cerebrales, fenómeno que se traduce en una onda de 15mms de agua en la curva de monitorización continua de la PIC.

   Las ondas cardiacas u ondas de pulso del LCR se deben primariamente a la contracción del ventrículo izquierdo. Aparece una onda de pulso inicial correspondiente a la sístole cardiaca (Onda de percusión, 1), seguida por una caída diastólica y una hendidura dicrota (E). Estas ondas del LCR semejan las del electrocardiograma.

   No obstante la forma exacta y amplitud de estas ondas dependen de la entrada arterial, la salida venosa y el estado del resto de los componentes intracraneales.

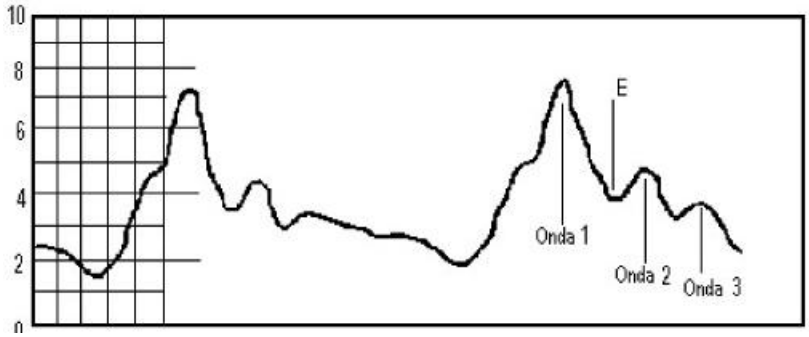



Figura 2.1.1. Ondas cardiacas de la PIC

2. La inspiración respiratoria, que aumenta la presión en el sistema venoso, también modifica la PIC y se evidencia en el gráfico a través de ondas de mayor amplitud, si la añadimos al pulso cardiaco puede alcanzar hasta 45 mms de agua.
   La fluctuación de la PIC con la respiración tiene una morfología más constante y está determinada por las variaciones en la presión de las cavidades abdominales y torácica, que a su vez modifican el retorno venoso a la aurícula derecha.

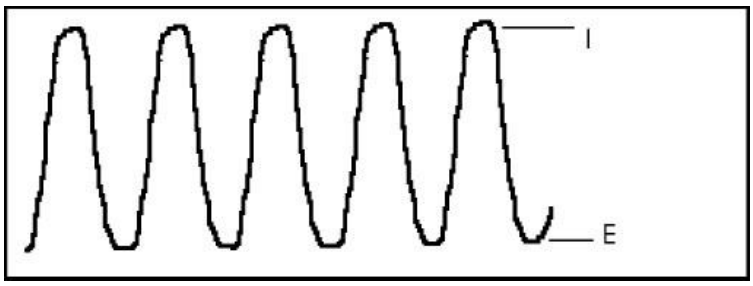



Figura 2.1.2. Ondas respiratorias. Valores en la inspiración (I) y expiración (E).

La superposición de ambas ondas en el mismo gráfico resultaría una imagen como la siguiente:

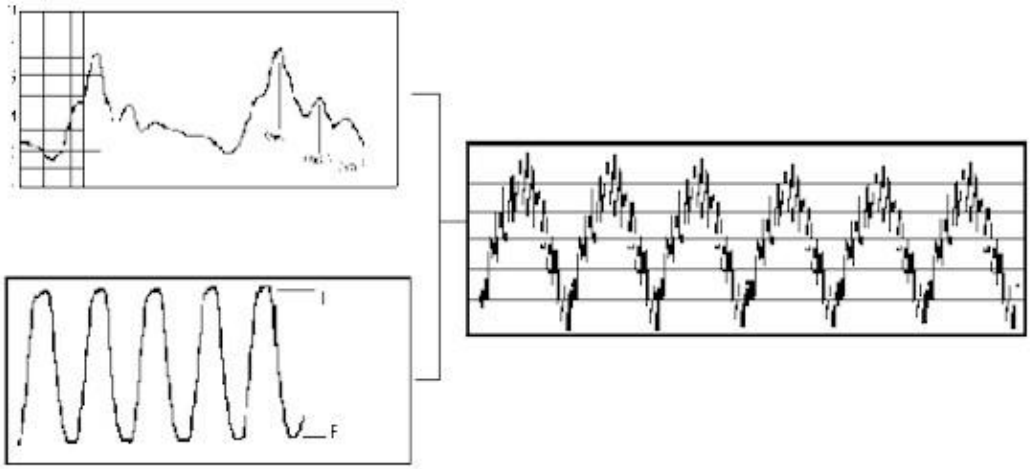



Figura 2.1.3. La gráfica de la PIC muestra ambas ondas al mismo tiempo, resultando el esquema de la derecha.

**CONCEPTOS A TENER EN CUENTA SOBRE LA MONITORIZACIÓN DE LA PIC.**

*REGISTROS NORMALES VS REGISTROS DE INFUSIÓN.*

- Registro normal: Se coloca el sensor en el ventrículo cerebral y se mide la presión a lo largo de unas 12-24 horas.
- Registro de infusión: Se infunde suero mediante una bomba de infusión constante dentro del ventrículo cerebral y mientras se mide la presión intracraneal. La infusión termina cuando:
    1. Sube la presión intracraneal por encima de 40
    2. El tiempo de infusión supera los 30 minutos
    3. La presión sube y se consigue una presión final estable

*EVENTOS.*

Un evento es una situación en la que el paciente no está en reposo, esto influye en el valor de la PIC y por tanto los valores recogidos en ese momento son poco fiables a la hora del análisis.

Como hemos mencionado en el apartado anterior la presión intracraneal se ve modificada por diversas situaciones fisiológicas. Algunos ejemplos de eventos en un paciente son:

- El paciente se sienta o se levanta, la PIC cambia con la postura.
- El paciente tose, vomita, o hace algún esfuerzo, la PIC cambia con estas maniobras (maniobras de valsalva).

*PARÁMETRO R.OUT*

La R.out es un parámetro que se mide en los registros de infusión. El líquido cefalorraquídeo, que rodea el cerebro, se forma de forma constante y se reabsorbe también de forma constante. La R.out es la resistencia a la reabsorción del líquido que se introduce con la bomba de infusión.

Para calcular esta resistencia son necesarios tres parámetros:

- Ritmo de infusión del líquido que se introduce con la bomba de infusión
- Valor de la basal: presión intracraneal basal al inicio del registro.
- Valor de la onda arriba: presión intracraneal al final del registro cuando se llega a un equilibrio.

*CALIBRACIÓN DE LA PANTALLA.*

Cada hora de monitorización de la PIC se debe representar en 20 cm de pantalla, ya que es un tamaño cómodo y en el cual se puede tener una idea general a simple vista.

El problema surge cuando en ocasiones se dispone de monitores más pequeños, para que este parámetro permanezca inalterable hay que ajustar el valor de un centímetro real en esas pantallas.

**EN QUÉ OCASIONES ES NECESARIO MONITORIZAR LA PIC.**

La monitorización de la presión intracraneal se realiza generalmente cuando hay sospecha de que alguno de los componentes de la cavidad craneana pueda aumentar su volumen.

- Las lesiones ocupantes de espacio, como los hematomas, tumores, abscesos y aneurismas y el edema, causado por traumatismos, aumentan el volumen cerebral. Mientras que las obstrucciones venosas causan un volumen sanguíneo aumentado.

- Asimismo, se puede hacer después de una cirugía para extirpar un tumor o reparar un daño a un vaso sanguíneo, si el equipo quirúrgico está preocupado respecto a una inflamación cerebral.

- Sospecha de hidrocefalia normotensa.

La monitorización de la presión intracraneal es crucial para la identificación del problema y para tratarlo de inmediato. Un aumento en la presión intracraneal significa que tanto los tejidos del sistema nervioso (neurales) como de los vasos sanguíneos (vasculares) están comprimidos.

Si esto se deja sin tratamiento, puede provocar un daño neurológico permanente y, en algunos casos, puede ser mortal.

**HIDROCEFALIA NORMOTENSA Y PRESIÓN INTRACRANEAL**

La hidrocefalia normotensa es una enfermedad cuyos síntomas son:

- Trastornos de la marcha (es el más característico)
- Deterioro mental
- Incontinencia urinaria (síntoma tardío)

Se da con más frecuencia en personas mayores de 60 años y con preponderancia ligera en pacientes de sexo masculino. Es el único tipo de demencia tratable y/o curable en la actualidad.

En su descripción inicial, el diagnóstico de esta enfermedad requería como criterio ineludible la obtención de un valor normal de la presión del líquido cefalorraquídeo (LCR) medida por punción lumbar. Sin embargo, la monitorización continua de la presión intracraneal (PIC) nos permite afirmar que la denominación de "hidrocefalia de presión normal" sólo se sustenta por la tradición, ya que el control continuo de la PIC ha demostrado que los pacientes afectados de este síndrome pueden presentar elevaciones transitorias o continuas de la PIC. Por ello, en la actualidad, la denominación más aceptada y extendida para hacer referencia a esta enfermedad es la de "hidrocefalia crónica del adulto", aunque recientemente se debate la posibilidad de que exista la hidrocefalia crónica de la infancia.

La causa de esta enfermedad quizá sea una insuficiente capacidad de absorción del LCR, pero el mecanismo exacto del desarrollo de los síntomas clínicos se desconocen.

Antes de pasar a realizar un tratamiento quirúrgico en estos pacientes, es necesario hacer un estudio minucioso para descartar una demencia senil.

**TRATAMIENTO:**

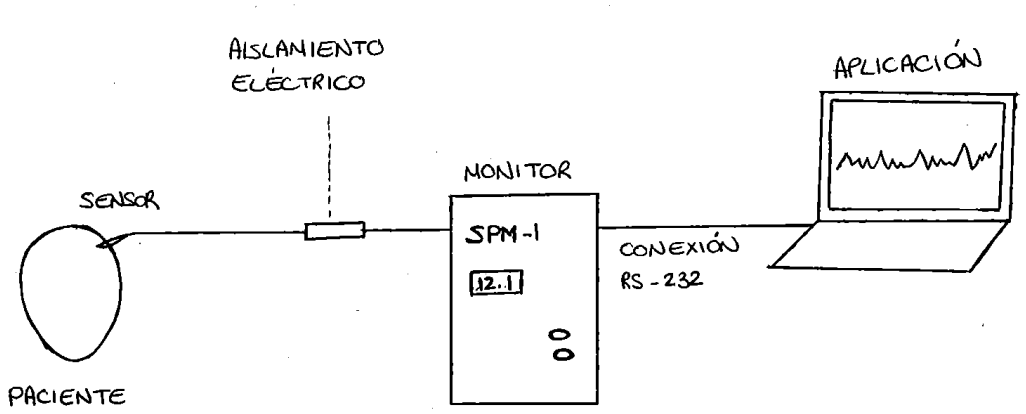
La presión intracraneal elevada se puede tratar:

- Drenando el LCR (líquido cefalorraquídeo) mediante un catéter.
- Cambiando el ajuste del ventilador (para las personas que se encuentren en un estado crítico, con un respirador artificial).
- Administrando ciertos medicamentos por vía intraven

## 2.2 CÓMO SE MIDE LA PIC

En la monitorización de la presión intracraneal se utiliza un dispositivo (sensor), colocado dentro de la cabeza del paciente, que percibe la presión dentro del cráneo y envía sus mediciones a otro dispositivo que las registra (monitor).

En general pueden dividirse en equipos acoplados a fluidos y aquellos que no usan líquidos como transmisión de señal de presión.




***SENSOR: Integra Neurosciences Camino, Sensor 110-4hm***

Los sensores se han desarrollado de la misma forma que han evolucionado los sitios de su colocación.

Los sensores iniciales estaban conectados a una columna de agua que era la encargada de trasmitir las variaciones de la presión, mas tarde fueron sistemas cerrados con balones sobre los cuales se ejercía la presión, todos con precauciones que evitaban la contaminación del LCR y de este modo hacían al sistema fiable en cuanto a los resultados y protegidos contra la sepsis (infección).

En la actualidad existen algunos muy sofisticados, como el que disponemos en este equipo, acoplados a un sistema de fibra óptica que transmite las modificaciones en la posición de un pequeño espejo y su consiguiente lectura electrónica. Por supuesto mientras más sofisticados sean el manejo y cuidado del equipo, exigen de personal más entrenado.

- *Dónde colocar el sensor*

  *El sitio más utilizado para medir la PIC son los ventrículos laterales. Este lugar ha permanecido como el mejor sitio para el estudio continuo de la PIC pues traduce con fiabilidad los aumentos de presión y permite además la evacuación de LCR durante los episodios hipertensivos.*

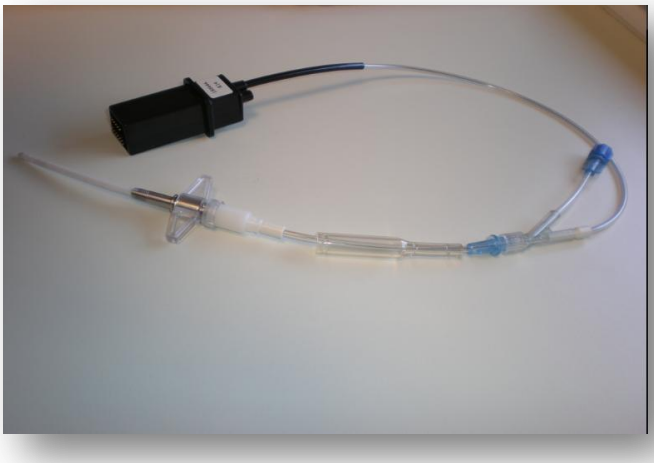


Figura 2.2.1. Integra Neuro Sciences Camino® Sensor 110-4hm

El sensor está eléctricamente aislado para evitar riesgos en el paciente.


Figura 2.2.2. Integra Neuro Sciences Camino® Aislamiento eléctrico.


Figura 2.2.3. Integra Neuro Sciences Camino® Sensor 110-4hm y aislamiento eléctrico.

***MONITOR:  Integra Neuro Sciences Camino, SPM-1 Single Parameter Monitor***

Es el encargado de recibir la señal del sensor, presentar el valor de la PIC instantáneo en una pequeña pantalla, y mandar una señal eléctrica al PC que contiene la aplicación.

El monitor se conecta al PC a través de una conexión por puerto serie RS-232.


Figura 2.2.4. Integra Neuro Sciences Camino® SPM-1 Single Parameter Monitor

***APLICACIÓN***

Es necesario analizar y almacenar los datos que el monitor proporciona en tiempo real, así pues, necesitamos una aplicación capaz de realizar todas las funciones requeridas por los médicos. Este equipo es un PC y la aplicación es el objeto de este proyecto.

El equipo ideal debe ser:

- Preciso en sus mediciones
- Seguro para el paciente
- Simple en su uso
- En lo posible de bajo coste económico.

### *2.2.1 Análisis de la aplicación anterior*

***FUNCIONALIDAD***

La aplicación actual permite:

- Representar una curva de PIC en tiempo real, es decir, durante una monitorización.
- Representar una curva de PIC de un registro antiguo

***LIMITACIONES***

La aplicación presenta varias limitaciones:

- NO distingue ficheros de datos recogidos de un registro normal o un registro de infusión.
- NO puede aumentar la escala para ver la curva de PIC en detalle, esto limita el estudio de ciertos patrones, como el de la hidrocefalia normotensa.
- NO distingue tramos de curva que hayan sucedido durante un evento.
- NO puede calcular el parámetro R.out
- NO es posible calibrar la pantalla.
- Los registros antiguos de pacientes (ficheros) están en una carpeta, sin ningún orden y con nombres que pone el personal sanitario manualmente.
  A la hora de ver un registro antiguo los especialistas simplemente buscan en esa carpeta, lo que se traduce en una tarea tediosa.

Estas limitaciones, obstaculizan un diagnóstico fácil para los especialistas, que pierden tiempo haciendo cálculos a mano, y representando ficheros de datos en Excel.

## 2.3 REQUISITOS

### *2.3.1 Requisitos de usuario*

En este apartado procederemos a enumerar los requisitos de usuario para la nueva aplicación.

*NOMENCLATURA REQUISTOS USUARIO*

Con el fin de distinguir las distintas clases de requisitos (usuario, software y hardware) utilizaremos una nomenclatura acorde a cada tipo.

En este caso los requisitos de usuario irán precedidos del prefijo *'RU'*, y seguidos de un número que corresponde a cada uno de los requisitos en particular.

- Los nombres de los requisitos de usuario quedarán de la siguiente manera:
  RU. *Número de requisito. Descripción del requisito*

*REQUISITOS DE USUARIO*

**RU.01 INTERFAZ GRÁFICA FÁCIL DE USAR**

El interfaz de usuario o interfaz gráfica es la parte del programa que permite a éste interactuar con él, y normalmente es el aspecto más importante de cualquier aplicación. Una interfaz gráfica sencilla e intuitiva es imprescindible para sacar el máximo rendimiento de la aplicación.

**RU.02 MONITORIZACIÓN DE LA PIC EN TIEMPO REAL**

La aplicación debe ser capaz de extraer los datos que manda el monitor de PIC en tiempo real y representarlos por pantalla.

**RU.03 REGISTROS DE INFUSIÓN SEPARADOS DE REGISTROS NORMALES**

En el proceso de la monitorización de la presión intracraneal se pueden llevar a cabo dos tipos de procedimientos:

- Registro normal: Se coloca el sensor en el ventrículo cerebral y se mide la presión a lo largo de unas 12-24 horas.
- Registro de infusión: Se infunde suero mediante una bomba de infusión constante dentro del ventrículo cerebral y mientras se mide la presión intracraneal. La infusión termina cuando:
  4. Sube la presión intracraneal por encima de 40.
  5. El tiempo de infusión supera los 30 minutos.

6. La presión sube y se consigue una presión final estable.

Es importante que los datos de los registros guardados (registros antiguos) estén clasificados según el tipo de registro que se realizó. Esta información siempre es imprescindible para el análisis de una monitorización.

### RU.04 POSIBILIDAD DE MARCAR EVENTOS MANUALMENTE

Un evento es una situación en la que el paciente no está en reposo, estos acontecimientos influyen sobre el valor de la presión intracraneal.

Marcar eventos manualmente implica que los especialistas puedan escribir el tipo de evento que está sucediendo, la hora de inicio y la hora de fin, y que los valores de PIC recogidos en ese intervalo de tiempo aparezcan diferenciados de los demás, ya que son tramos que normalmente se descartan al hacer el análisis de la curva, por ser tramos poco fiables.

Algunos ejemplos de eventos son:

- El paciente se sienta o se levanta, ya que la PIC cambia con la postura.
- El paciente tose, vomita, o hace algún esfuerzo, ya que la PIC cambia con estas maniobras (maniobras de valsalva)

### RU.05 POSIBILIDAD DE MARCAR BASAL, ONDA ARRIBA Y CALCULAR R.OUT SOBRE LA GRÁFICA

Hay ciertos parámetros que el personal sanitario debe conocer para realizar un correcto análisis de la curva de PIC.

En particular, este requisito está centrado en la R.out, parámetro que se mide en los registros de infusión. La R.out es la resistencia a la reabsorción del líquido que se introduce con la bomba de infusión.

Para calcular esta resistencia es necesario poder marcar sobre la curva la basal y la onda arriba, es decir la presión intracraneal basal al inicio del registro y la presión intracraneal al final del registro cuando se llega a un equilibrio.

### RU .06 POSIBILIDAD DE AUMENTAR/DISMINUIR LA GRÁFICA DE LA PIC : DETECCIÓN DE HIDROCEFALÍA NORMOTENSA

Para analizar una curva de PIC, y en concreto detectar patrones de hidrocefalia normotensa es necesario que el especialista pueda aumentar/disminuir la gráfica de PIC según sus conveniencias.

Este aumento/disminución se tratará en los ejes, temporal y de amplitud, independientemente. Siendo posible de esta manera un amplio abanico de posibilidades de vistas de la gráfica.

Los niveles de aumento necesarios en ambos ejes han sido proporcionados por el especialista colaborador en el proyecto.

- ❖ Vistas del eje temporal:
  - 10 cm / hora
  - 20 cm/hora
  - 40 cm/ hora
  - 80 cm /hora
- ❖ Vistas del eje valor de la PIC
  - Aumento 0 (valores de 0 a 80 visibles simultáneamente en la pantalla)
  - Aumento 1 (valores de 0 a 80 espaciados con precisión de 5 decimales, visibles simultáneamente 40 valores)
  - Aumento 2 (valores de 0 a 80 espaciados con precisión de 1 decimal, visibles simultáneamente 20 valores) )

### RU.07 POSIBILIDAD DE CALIBRAR LA PANTALLA SEGÚN EL MONITOR DEL PC

Uno de los requisitos de usuario consiste en que en la gráfica de la PIC se represente cada hora de monitorización en 20 cm de pantalla, ya que es un tamaño cómodo y en el cual se puede tener una idea general a simple vista.

El problema surge porque en ocasiones se dispone de monitores más pequeños y para que este parámetro permanezca inalterable hay que ajustar el valor de un centímetro real en esas pantallas.

Para solucionar este problema, el usuario ha pedido que sea posible ajustar el valor de un centímetro, de forma que baste con colocar una regla sobre la pantalla y marcar en una barra de desplazamiento la medida de un centímetro real.

### RU.08 POSIBILIDAD DE REGISTRAR NUEVOS PACIENTES Y VER PACIENTES REGISTRADOS

La aplicación debe ser capaz tanto de registrar nuevos pacientes, como de poder visualizar los pacientes que ya han sido ingresados en el sistema.

**RU.09 POSIBILIDAD DE ABRIR REGISTROS ANTIGUOS PARA VER LA EVOLUCIÓN DEL PACIENTE**

La aplicación debe ser capaz de, una vez seleccionado un paciente antiguo, mostrar sus todos sus registros de PIC.

El usuario podrá elegir el registro que deseé abrir y este se abrirá en una pantalla, donde se representará la curva de PIC en un entorno óptimo para proceder a su análisis.

## *2.3.2 Requisitos de Software y Hardware*

En este apartado procedemos a enumerar los requisitos de software y hardware necesarios para el funcionamiento de la nueva aplicación.

*NOMENCLATURA REQUISTOS SOFTWARE Y HARDWARE*

Con el fin de distinguir las distintas clases de requisitos utilizaremos una nomenclatura acorde a cada tipo.

En este caso los requisitos de software irán precedidos del prefijo *'RS'*, y seguidos de un número que corresponde a cada uno de los requisitos en particular.

RU. *Número de requisito. Descripción del requisito*

Los requisitos de hardware irán precedidos del prefijo *'RH'*, y seguidos de un número que corresponde a cada uno de los requisitos en particular.

RU. *Número de requisito. Descripción del requisito*

*REQUISITOS DE SOFTWARE*

Tenemos una limitación en cuanto a la instalación de nuevo software en el PC del hospital, no debemos instalar nada que conlleve mantenimiento y en términos generales instalar lo mínimos posible.

Debemos tener en cuenta esta limitación a la hora de tomar las decisiones de diseño de la nueva aplicación.

- **RS.01 SISTEMA OPERATIVO WINDOWS XP**
- **RS.02 *JAVA RUNTIME ENVIRONMENT (JRE)***

  El JRE es el entorno mínimo para ejecutar programas Java .
- **RS.03 *BIBLIOTECA DE COMUNICACIÓN POR PUERTO SERIE DE JAVA***

  Biblioteca Giovynet

*REQUISITOS DE HARDWARE*

- **RH.01 ORDENADOR QUE CONTIENE LA APLICACIÓN**

  Procesador Intel Pentium 3.

  Puerto serie, rs-232.

- **RH.02 CONEXIÓN PC-MONITOR**
  Cable serie cruzado.

- **RH.03 MONITOR : INTEGRA NEURO SCIENCES CAMINO, SPM-1 SINGLE PARAMETER MONITOR**

- **RH.04 SENSOR : INTEGRA NEUROSCIENCES CAMINO, 110-4HM**

  Equipo para la monitorización de la presión micro-tornillo con ventricular.

# 3. DISEÑO Y DESARROLLO DE LA APLICACIÓN

# INTRODUCCIÓN

Establecidos los requisitos de usuario que tenemos que cumplir, y analizado el entorno software y hardware del que disponemos, procedemos a tomar las decisiones de diseño para la nueva aplicación.

Comenzamos diseñando la estructura de la aplicación, para que sea lo más sencilla y funcional posible.

- 3.1 DISEÑO DE LA APLICACIÓN
  - *3.1.1 Estructura de la aplicación*
  - *3.1.2 Almacenamiento de datos*

Elegimos un entorno de desarrollo óptimo.

- 3.2 ENTORNO DE DESARROLLO
  - *3.2.1 Lenguaje de programación*
  - *3.2.2 Entorno de programación*

Por último, procedemos al desarrollo de la aplicación. En primer lugar debemos implementar una interfaz gráfica, y hecho esto, dotaremos a la interfaz de funcionalidad de modo que se cumplan todos los requisitos de usuario.

- 3.3 DISEÑO GRÁFICO Y FUNCIONAL DE LA APLICACIÓN

Una vez terminada la aplicación pasaremos a la fase de pruebas, que está recogida en el siguiente apartado.

## 3.1 ESTRUCTURA DE LA APLICACIÓN

### 3.1.1 Estructura de la aplicación

La aplicación consta de cuatro pantallas que permiten representar y analizar la curva de presión intracraneal (PIC) de una monitorización actual o un archivo antiguo.

Cada una de estas pantallas está dedicada a un propósito, y tiene unas funciones específicas en base a cubrir los requisitos de usuario (RU.XX, ver apartado 2.3.1).

**Pantalla 1. Pantalla de inicio.**

- Función 1.1 Acceso protegido con contraseña.

**Pantalla 2. Pantalla de registro y selección de pacientes.**

- Función 2.1 Buscador de pacientes. (RU.08)
- Función 2.2 Inicio de monitorización. (RU.02)
- Función 2.3 Ver registros antiguos. (RU.09)
- Función 2.4 Calibrar pantalla. (RU.07)

**Pantalla 3. Pantalla de monitorización de la PIC.**

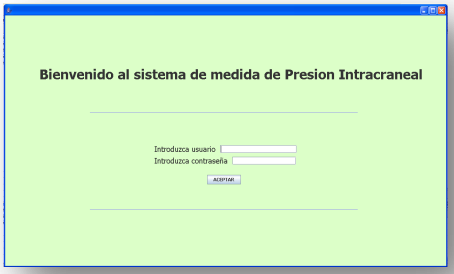
- Función 3.1 Monitorización de la PIC. (RU.02)
- Función 3.2 Registros normales separados de registros de infusión. (RU.03)
- Función 3.3 Marcar manualmente eventos. (RU.04)
- Función 3.4 Finalizar monitorización. (RU.02)
- Función 3.5 Acceder a la pantalla de estudio de la curva para el registro actual. (RU.02)
- Función 3.6 Acceder a la pantalla de estudio de la curva para un registro antiguo. (RU.09)

**Pantalla 4. Pantalla de estudio de la curva.**

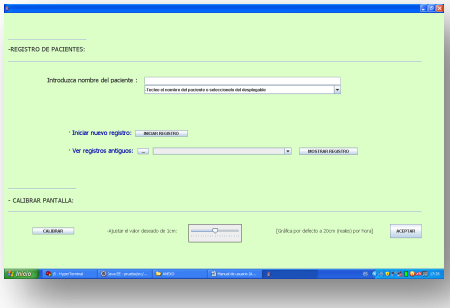
- Función 4.1 Mostrar el registro seleccionado. (RU.09)
- Función 4.2 Aumentar/Disminuir la escala de la PIC. (RU.06)
- Función 4.3 Aumentar/Disminuir la escala temporal. (RU.06)
- Función 4.4 Marcar basal/ onda arriba. (RU.05)
- Función 4.5 Calcular R.out (RU.05)
- Función 4.6 Ver fichero de eventos. (RU.04)

Las pantallas de la aplicación se suceden de la siguiente manera según las opciones de usuario:

1-Pantalla de bienvenida.

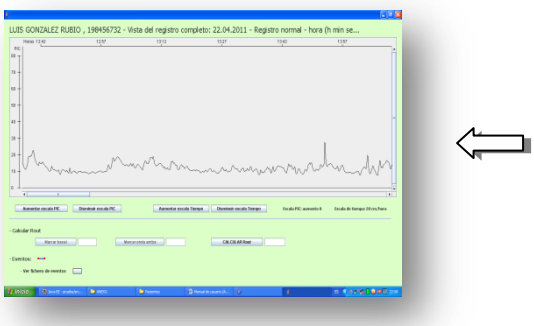


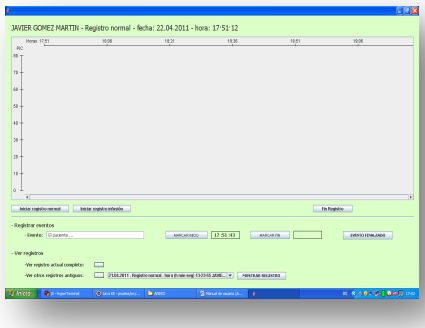

2-Pantalla de registro de pacientes.




4-Pantalla de estudio de la curva

3-Pantalla de monitorización de la PIC




**En el apartado 3.3, Diseño gráfico y funcional, veremos esto en detalle.**

### 3.1.2 Almacenamiento de datos

Antes de diseñar una aplicación que se va a utilizar en diferentes pacientes, y cuyos datos deben quedar registrados tenemos que pensar en cómo vamos a ordenar y organizar esos datos.

Crear una base de datos para este propósito no es la opción más apropiada ya que implica instalaciones y mantenimiento, que en este proyecto estamos intentando evitar (ver 2.3.2 Requisitos de software y hardware).

El almacenamiento de ficheros de la aplicación actual se hace en un directorio al que los especialistas tienen acceso sin necesidad de abrir la aplicación. Se ha decidido seguir este método de almacenamiento, añadiendo dos grandes mejoras:

1. La primera es la creación de un directorio para cada paciente, al contrario que en la aplicación actual dónde existe una única carpeta general en la cual se almacenan los registros de todos los pacientes juntos.

    - Si el personal sanitario quiere ver un registro antiguo debe buscar en esa gran carpeta el archivo que desea abrir, lo cual es una tarea muy tediosa.

2. La segunda mejora es que esta aplicación nombra los registros de forma automática (nombre de paciente-fecha) quedando así todos bien clasificados y nombrados, y haciendo posible las búsquedas desde la misma aplicación tecleando las primeras letras del nombre del paciente, y seleccionando de un desplegable sus registros a visualizar.

    - En la aplicación actual esto es imposible ya que los nombres de cada registro los pone el médico manualmente en el momento de la monitorización, y por tanto sin ninguna regla que permita una búsqueda automática por parte de la aplicación.

**ESTRUCTURA DEL DIRECTORIO DE PACIENTES**

Como hemos dicho, todos los pacientes están registrados en un Directorio General de Pacientes, en el que cada paciente tiene su propio directorio principal, junto con dos subdirectorios, para almacenar archivos de esta forma:

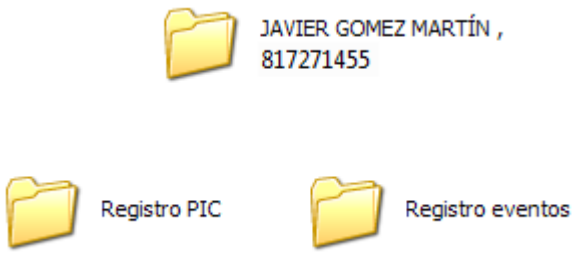



Figura 3.1.1. Directorio de un paciente.

El nombre del directorio principal está formado por el nombre del paciente y su número de historia, para solucionar posibles ambigüedades en los nombres.

Cada directorio de paciente contiene dos subdirectorios, uno para guardar los registros de la PIC, y otro para guardar los ficheros de eventos asociados.

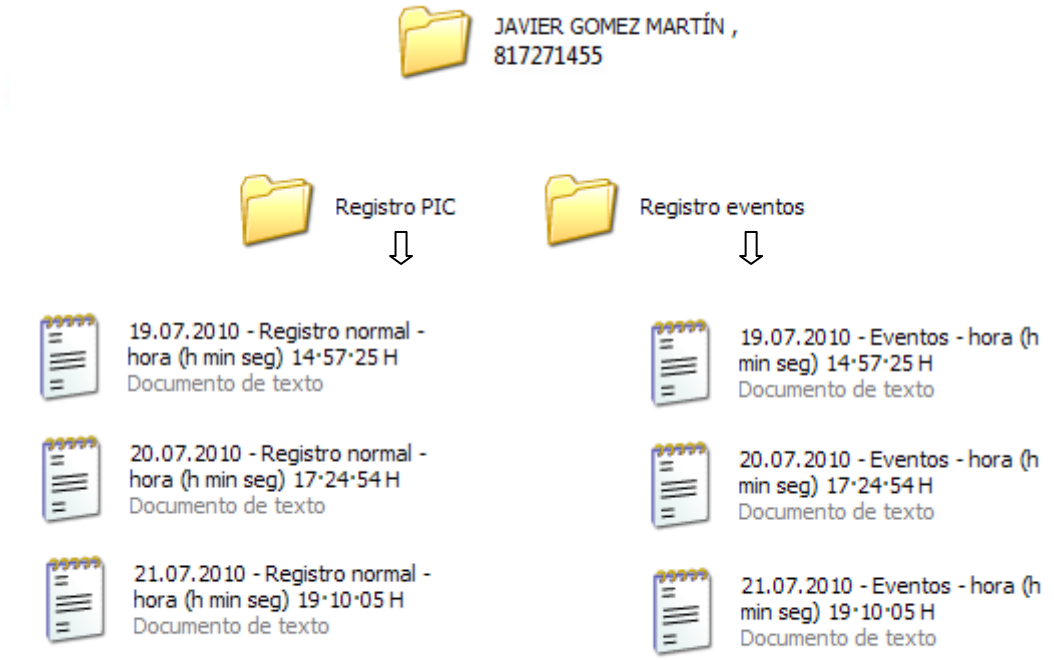



Figura 3.1.2. Registros de un paciente.

Los ficheros de eventos están asociados a cada fichero de PIC y llevan información de los eventos ocurridos durante esa monitorización.

### ➔ Registros de PIC

- Cada vez que se inicia una monitorización en tiempo real de la PIC se crea un fichero de PIC.
  Los registros de PIC guardan los valores de la presión intracraneal monitorizada en cada instante y se nombran de la siguiente manera, según sean registros normales o de infusión:

  fecha - Registro normal - hora (h min seg) - *nombrePaciente* .txt
  fecha - Registro infusión - hora (h min seg) - *nombrePaciente* .txt

  Ejemplo: 22.06.2011 - Registro normal – hora (h min seg) 12·22·11 JAVIER GOMEZ MARTIN

  De esta forma cumplimos el requisito RU.03 de tener diferenciados los registros normales de los de infusión.

- Contenido:

| Fecha | Hora | PIC | Hay evento (0-no, 1-si) |
|---|---|---|---|
| 18/06/2011 | 04:40:21 | 3.1 | 0 |
| 18/06/2011 | 04:40:23 | 3.1 | 0 |
| 18/06/2011 | 04:40:24 | 3.1 | 0 |
| 18/06/2011 | 04:40:25 | 3.1 | 0 |
| 18/06/2011 | 04:40:26 | 3.1 | 0 |
| 18/06/2011 | 04:40:28 | 3.1 | 0 |

### ➔ Registros de Eventos

- Por cada fichero de PIC creado se creará un fichero de Eventos, donde se recogerán los datos de los eventos ocurridos durante esa monitorización de la PIC.

  Cada registro de eventos está asociado a su registro PIC correspondiente mediante la fecha, hora de comienzo y nombre del paciente:

  fecha - Eventos - hora (h min seg) -*nombrePaciente* .txt

  Ejemplo: 22.06.2011 - Eventos - hora (h min seg) 12·22·11 JAVIER GOMEZ MARTIN

- Contenido:

```
El paciente (acción 1)   INICIO  17 :07 :13   FIN  17 :07 :47
-
El paciente (acción 2)   INICIO  17 :10 :19   FIN  17 :11 :27
-
```

Este directorio se creará en el momento de la instalación de la aplicación y tendrá un determinado *PATH* que la aplicación debe saber para realizar las búsquedas. Si en algún momento se cambiara de ubicación el directorio habría que actualizar la aplicación si queremos que funcione correctamente.

**ACCESO A DATOS DEL DIRECTORIO DE PACIENTES.**

Hay partes de la aplicación que necesitan extraer datos del Directorio de Pacientes, vamos a concretar en qué momentos es necesario el acceso a éste.

- Pantalla 1: no necesita acceder al Directorio de Pacientes.

- Pantalla 2: Necesita acceder al Directorio de Pacientes para realizar las funciones:

    - Función 2.1 *Buscador de pacientes*
      Accede en el momento en que se empieza a teclear un nombre de paciente, hace una búsqueda en el directorio y presenta los nombres registrados que más coinciden con lo tecleado en un desplegable.

    - Función 2.3 *Ver registros antiguos*
      Busca en el directorio los registros antiguos del paciente seleccionado y los presenta en un desplegable.

- Pantalla 3: Accede al Directorio de Pacientes para realizar las funciones:

    - Función 3.1 *Monitorización de la PIC.*
      Accede en el momento en que comienza a escribir los datos recibidos en tiempo real en un fichero del directorio.

    - Función 3.2 *Registros normales separados de registros de infusión.*
      Accede en el momento de iniciar una monitorización, al crear los nuevos ficheros físicos correspondientes al registro actual.

    - Función 3.3 *Marcar manualmente eventos.*
      Debe acceder para escribir los eventos en el fichero correspondiente.

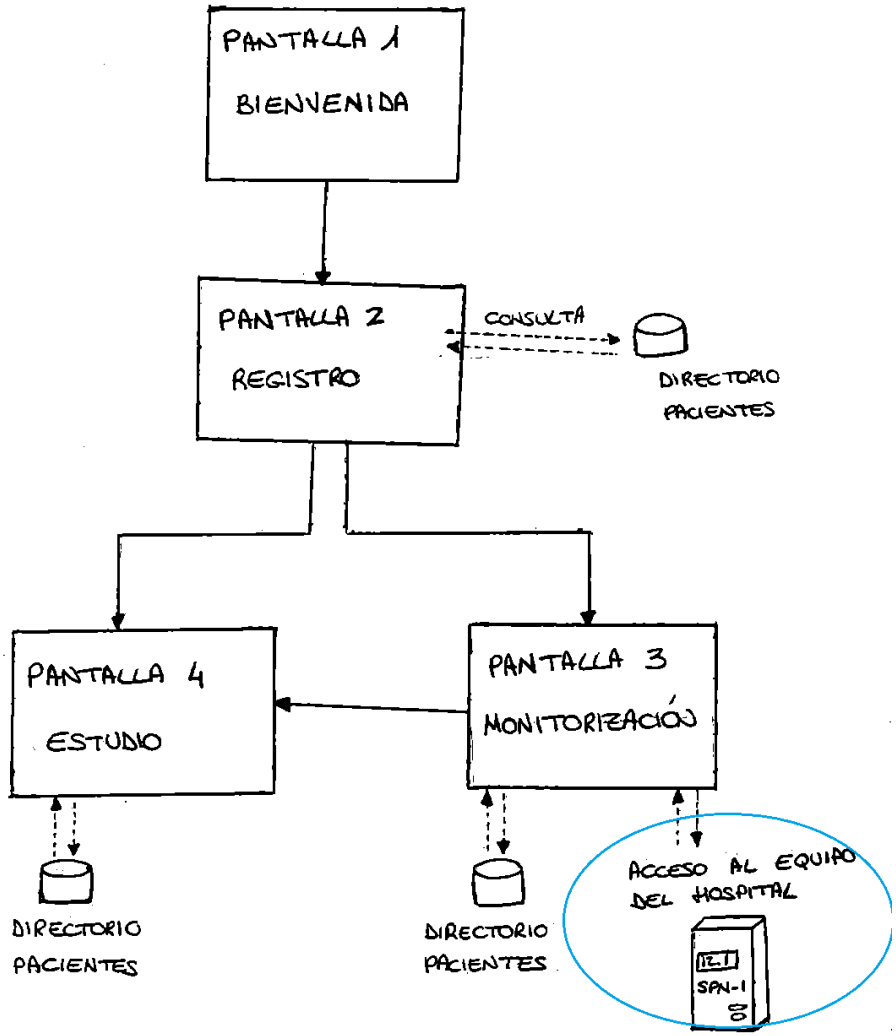
    - Función 3.6 *Acceder a la pantalla de estudio de la curva para un registro antiguo.*
      Busca en el directorio los registros antiguos del paciente actual y los presenta en un desplegable.

- Pantalla 4: Accede al Directorio de Pacientes para realizar las siguientes funciones:

    - Función 4.1 *Mostrar el registro seleccionado.*
      Debe acceder para leer el registro seleccionado y mostrarlo por pantalla.

    - Función 4.6 *Ver fichero de eventos.*
      Debe acceder para leer el fichero de eventos correspondiente.

**ESQUEMA DE ACCESOS A DATOS DE LA APLICACIÓN.**




**ACCESO A DATOS DE LA PIC POR PUERTO SERIE (ACCESO AL EQUIPO DEL HOSPITAL).**

Esta función es exclusiva de la pantalla 3 o pantalla de monitorización de la PIC. En esta ocasión los datos que se representan no son leídos de un fichero sino que se reciben por una conexión puerto serie.

En este momento es cuando utilizamos el API java de comunicaciones, con el que habilitamos un puerto serie para la comunicación con el monitor del hospital. También establecemos los parámetros de la comunicación (velocidad, etc…), y vamos recibiendo los datos del sensor en tiempo real. Estos datos son los que se irán representando en la pantalla de monitorización (pantalla 3) que por tanto es una pantalla dinámica.

## 3.2 ENTORNO DE DESARROLLO

### 3.2.1 Lenguaje de programación

La elección del lenguaje adecuado de programación es importante, debemos buscar un equilibrio entre las características funcionales del lenguaje con aspectos como el control y complejidad de este.

En una aplicación del tipo que estamos creando tenemos que satisfacer principalmente dos necesidades:

- Interfaz de usuario práctica e intuitiva.
- Capacidad de recibir datos y mostrarlos en tiempo real.

Las opciones planteadas que cumplen estos requisitos son:

- Visual Basic: no es conveniente, pues implica usar componentes instalados en el ordenador y como hemos mencionado antes (2.3 Requisitos de software) es preferible instalar lo menos posible.
- C, C++ : óptimo
- Java : óptimo

Tanto java como C se ajustan a los requisitos de manera similar, finalmente el lenguaje elegido ha sido java por tener el desarrollador más conocimiento sobre éste.

**LENGUAJE JAVA**

Java se creó como parte de un proyecto de investigación, en un momento en el que C++ era el lenguaje del momento. Pero a lo largo del tiempo, las dificultades encontradas con C++ crecieron hasta el punto en que se pensó que los problemas podrían resolverse mejor creando una plataforma de lenguaje completamente nueva.

Se extrajeron decisiones de diseño y arquitectura de una amplia variedad de lenguajes como Eiffel, SmallTalk, Objetive C y Cedar/Mesa. El resultado es un lenguaje que se ha mostrado ideal para desarrollar aplicaciones de usuario, y que además soporta varios hilos de ejecución (multitarea).

***CARACTERÍSTICAS DE JAVA:***

- Lenguaje orientado a objetos.
- Sintaxis inspirada en la de C/C++.
- Lenguaje multiplataforma: Los programas Java se ejecutan sin variación (sin recompilar) en cualquier plataforma soportada (Windows, UNIX, Mac...)
- Lenguaje interpretado: El intérprete a código máquina (dependiente de la plataforma) se llama Java Virtual Machine (JVM). El compilador produce un código intermedio independiente del sistema denominado Java *bytecode*.

- Creado por SUN Microsystems, que distribuye gratuitamente el producto base, denominado JDK (Java Development Toolkit) o actualmente J2SE (Java 2 Standard Edition).
- API distribuida con el J2SE muy amplia. Código fuente de la API disponible.

## QUÉ INCLUYE EL J2SE (JAVA 2 STANDARD EDITION)

***HERRAMIENTAS PARA GENERAR PROGRAMAS JAVA.***

Compilador, depurador, herramienta para documentación, etc.

***CÓDIGO FUENTE DE LA API (OPCIONAL).***

***DOCUMENTACIÓN.***

***JRE (JAVA RUNTIME ENVIRONMENT)***.

El JRE es el entorno mínimo para ejecutar programas Java 2.

En aquellos sistemas donde se vayan a ejecutar programas Java, pero no compilarlos, el JRE es suficiente (Este es el caso del PC del hospital).

El JRE incluye:

- La JVM (Java Virtual Machine)

La JVM es una máquina virtual capaz de interpretar y ejecutar instrucciones expresadas en un código binario especial (el Java bytecode, generado por el compilador del lenguaje Java). La JVM depende de la plataforma que estemos utilizando.

La gran ventaja de la máquina virtual java es aportar portabilidad al lenguaje de manera que existen diferentes máquinas virtuales java para diferentes arquitecturas. Así un programa .class escrito en un Windows puede ser interpretado en un entorno Linux. Tan solo es necesario disponer de dicha máquina virtual para dichos entornos.

- El API de Java.

El API Java es una Interfaz de Programación de Aplicaciones (API: por sus siglas en inglés) provista por los creadores del lenguaje Java, y que da a los programadores los medios para desarrollar aplicaciones Java.

Como el lenguaje Java es un Lenguaje Orientado a Objetos, el API de Java provee de un conjunto de clases utilitarias para efectuar toda clase de tareas necesarias dentro de un programa

En la comunidad de desarrollo Java se suele identificar cada una de las diferentes bibliotecas existentes como API's java. Cuando se construye un sistema informático este suele emplear diversas API`s.

El API de Java es vastísimo. En la aplicación se han utilizado las siguientes bibliotecas (ó API's):

| Package o grupo de packages | Descripción |
|---|---|
| java.applet | Proporciona las clases necesarias para crear applets, así como las clases |

| | |
|---|---|
| | que usa ul applet para comunicarse con su contexto. |
| java.awt.* | El AWT (Abstract Windows Toolkit) proporciona el entorno base para todas las clases de manipulación del interfaz gráfico del usuario. El AWT apareció en la versión 1.0 del JDK y fue parcialmente sustuituido y sustancialmente mejorado en la versión 1.1 (con el conjunto de componentes conocido como swing). Actualmente se mantiene porque es la base del swing aunque muchos de sus elementos ya no se usan. |
| java.io | Es un package fundamental. Proporciona los mecanismos para las operaciones de entrada/salida de flujos de datos (streams), así como la serialización (capacidad de los objetos para ser transformados en flujos de datos), y soporte para los sistemas de archivos. |
| java.lang.* | Proporciona clases básicas para cualquier programa. (Threads, clases envoltorio, entrada/salida/error estándar, etc.). Asi como clases que proporcionan la caracteristica de 'reflexión', es decir la capacidad de las clases de averigurar como están construidas ellas mismas u otras clases. |
| Java.math | Proporciona clases para realiza cálculos aritméticos de cualquier preción. Así como funciones matemáticas generales (Trigonometría, aleatorización, etc.) |
| java.security.* | Clases que implantan el esquema de seguridad de Java. |
| java.text | Proporciona clases e interfaces para la manipulación de texto, fechas, números y mensajes de una forma independiente del idioma. |
| java.util.* | Contiene la 'collections framework', o conjunto de clases para manipulación de conjuntos de objetos (colas, pilas, listas, diccionarios, árboles, tablas hash, etc.). Además tiene varios conjunts de utilidades para manipulación de fecha y hora, generación de números aleatorios y manipulación de ficheros comprimidos en formato ZIP y JAR. El formato JAR (Java ARchive) es una extensión del formato ZIP que permite empaquetar clases java compiladas para su ejecución. |
| javax.accesibility | Contiene un conjunto de clases para las tecnologías de asistencia relacionadas con los interfaces gráficos. |
| javax.swing.* | Conjunto extenso de clases para el configuación del interface gráfico de usuario. Reemplaza parcialmente al AWT. Su característica más importante es que es independiente de la plataforma. |

## BIBLIOTECA DE COMUNICACIONES EN JAVA : GIOVYNET

El PC está conectado al monitor del hospital mediante un puerto RS232, y por tanto, necesitamos una biblioteca que nos facilite la recepción de datos recogidos por el sensor.

El API javax.comm fue la biblioteca que Sun desarrolló para realizar comunicaciones en java por puerto serie, pero esta versión comenzó a dar problemas y ya no tiene soporte para Windows. La han sustituido otras bibliotecas, entre las que destacan Giovynet y RXTX, ambas similares en prestaciones. La elegida para este proyecto es Giovynet.

Giovynet provee a las aplicaciones de acceso al hardware RS232 (puertos serie) y acceso limitado a IEEE-1284 (puertos paralelo).

### *CARACTERÍSTICAS DE GIOVYNET*

- Enumeración de los puertos (se puede configurar la asignación de puertos)
- Configuración del puerto (velocidad de transmisión, bits de parada, paridad)
- Transferencia de datos a través de puertos RS-232
- Opciones de control de flujo hardware y software

## BIBLIOTECAS GRÁFICAS EN JAVA

El interfaz de usuario o interfaz gráfica es la parte del programa que permite a éste interactuar con él, y normalmente es el aspecto más importante de cualquier aplicación. Las interfaces de usuario pueden adoptar muchas formas, que van desde la simple línea de comandos hasta las interfaces gráficas que proporcionan las aplicaciones.

Una aplicación sin un interfaz fácil, impide que los usuarios saquen el máximo rendimiento del programa. Java proporciona los elementos básicos para construir interfaces gráficas a través de la biblioteca AWT, y opciones para mejorarlas mediante Swing.

### *JAVA FOUNDATION CLASSES*

Las JFC (Java Foundation Classes) son parte del API de Java, compuesto por clases que sirven para crear interfaces gráficas para las aplicaciones y *applets* de Java.

Las JFC constan de las siguientes API principales:

- AWT.
- Swing.
- Java 2D.
- Drag-and-Drop.
- Accessibility.

Las JFC contienen dos paquetes gráficos: AWT y Swing.

- AWT presenta componentes pesados, que en cada plataforma sólo pueden tener una representación determinada. Está disponible desde la versión 1.1 del JDK como *java.awt*.

- Swing presenta componentes ligeros, que pueden tomar diferentes aspecto y comportamiento ya que lo toman de una biblioteca de clases. Está disponible desde la versión 1.2 del JDK como *javax.swing*
  Aunque Swing esté separado del AWT, sus componentes utilizan la infraestructura de AWT, incluyendo el modelo de eventos AWT. Por eso, la mayoría de los programas Swing necesitan importar dos paquetes AWT: *java.awt.** y *java.awt.event.**.

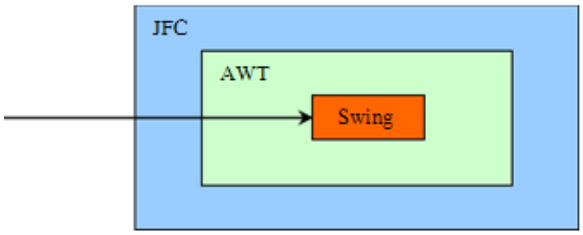



Figura . Relación existente entre Swing, el AWT, y las JFC.

Como regla, los programas no deben usar componentes pesados de AWT junto a componentes Swing, ya que los componentes de AWT son siempre pintados sobre los de Swing.

***VENTAJAS PAQUETE SWING RESPECTO A SU ANTECEDENTE AWT***

- Amplia variedad de componentes: En general las clases que comiencen por "J" son componentes que se pueden añadir a la aplicación (ej: *JButton)*.
  Las clases de Swing se parecen mucho a las de AWT. De hecho todas las clases de AWT tienen una nueva versión en Swing con el prefijo *J*. Es decir, la clase *Panel* de AWT tiene una clase *JPanel* en Swing.
- Swing incorpora nuevos gestores de impresión, ampliando los cinco que AWT incorporaba
- Aspecto modificable (*look and feel*): Se puede personalizar el aspecto de las interfaces o utilizar aspectos que existen por defecto (Metal Max, Windows, etc).
- Contenedores anidados: Cualquier componente puede estar anidado en otro. Por ejemplo, un gráfico se puede anidar en una lista.
- Bordes complejos: Los componentes pueden presentar nuevos tipos de bordes. Además el usuario puede crear tipos de bordes personalizados.
- Diálogos personalizados: Se pueden crear multitud de formas de mensajes y opciones de diálogo con el usuario, mediante la clase *JOptionPane*.
- Clases para diálogos habituales: Se puede utilizar *JFileChooser* para elegir un fichero, y *JColorChooser* para elegir un color.
- Componentes para tablas y árboles de datos: Mediante las clases *JTable* y *JTree*.
- Potentes manipuladores de texto: Además de campos (JTextFiled) y áreas de texto (JTextArea), se presentan campos de sintaxis oculta *JPassword*, y texto con múltiples fuentes *JTextPane*. Además hay paquetes para utilizar ficheros en formato HTML o RTF.

Ya que esta aplicación debe continuar en pruebas hasta su instalación final en el hospital se ha incluido cierta documentación de utilidad sobre esta biblioteca para futuros programadores. Para más información ver Anexo, Interfaz gráfica con swing.

## 3.2.2 Entorno de desarrollo

El sistema operativo en el que he desarrollado la aplicación es Windows 7.

El entorno necesario para escribir, compilar y ejecutar esta aplicación Java es el siguiente:

- J2SE (Java 2 Standard Edition)
- Entorno de Desarrollo Integrado para Java (IDE)

### J2SE (JAVA 2 STANDARD EDITION)

El J2SE incluye el compilador Java, el entorno de tiempo de ejecución (JRE Java Runtime Environment) y varias herramientas de ayuda (como se ha explicado en el apartado 3.1.1 *'Lenguaje'*).

He descargado la versión jdk 1.6 y añadido una biblioteca de comunicaciones (Giovynet) que permite a la aplicación recibir datos de un puerto serie.

### ENTORNO DE DESARROLLO INTEGRADO PARA JAVA (IDE): ECLIPSE

Un IDE, Integrated Development Environment o Entorno de Desarrollo Integrado, es un programa informático compuesto por un conjunto de herramientas de programación que sirven como ayuda para desarrollar aplicaciones o componentes.

Un IDE consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (GUI). Puede dedicarse en exclusiva a un sólo lenguaje de programación o bien, poder utilizarse para varios.

Un entorno que se ajusta bien a nuestras necesidades es Eclipse, que tiene interesantes utilidades para Java (plugin para interfaces gráficas) y además es gratuito.

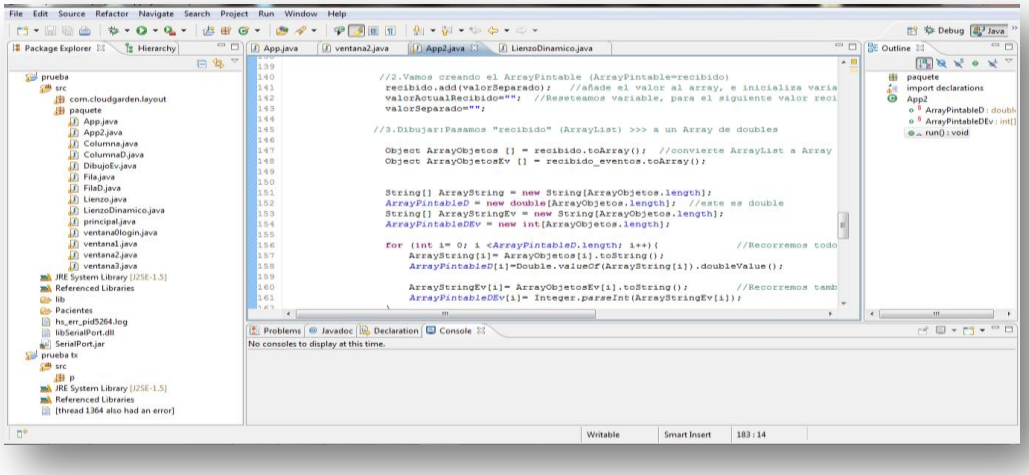


Figura 3.2.1.Entorno de desarrollo: Eclipse

***CARACTERÍSTICAS DE ECLIPSE***

- *IDE universal:* No solo es para java, también se puede usar con C, C++, cobol, D, Tex, etc.
- *Plugins:* Se le pueden añadir multitud de plugins para hacer de todo, desde plugins para crear interfaces graficas (Visual Editor), hasta descompiladores (jadclipse).
- El consumo de memoria es algo inferior a Netbeans aunque con la última versión de este último no hay tanta diferencia.
- Existen versiones para casi cualquier software, Windows y Linux entre ellos.
- Se puede traducir al castellano (y muchas otras lenguas): Solo hay que descargar las traducciones cedidas por IBM. Estan en http://download.eclipse.org/ sólo hay que buscar el enlace Language_Packs. Estos packs además tienen toda la documentación y la ayuda traducidas.

***PLUG-IN PARA CREAR INTERFACES GRÁFICAS EN ECLIPSE: JIGLOO***

Un plug-in es un módulo de hardware o software que añade una característica o un servicio específico a un sistema más grande.

La idea es que el nuevo componente se *enchufa* simplemente al sistema existente, aumentando así las funcionalidades del programa principal.

Jigloo, es un constructor de interfaces Swing para Java, compatible con Eclipse, y que facilita la creación de componentes y su colocación dentro de la ventana de la interfaz.

Web oficial: http://www.cloudgarden.com/jigloo/

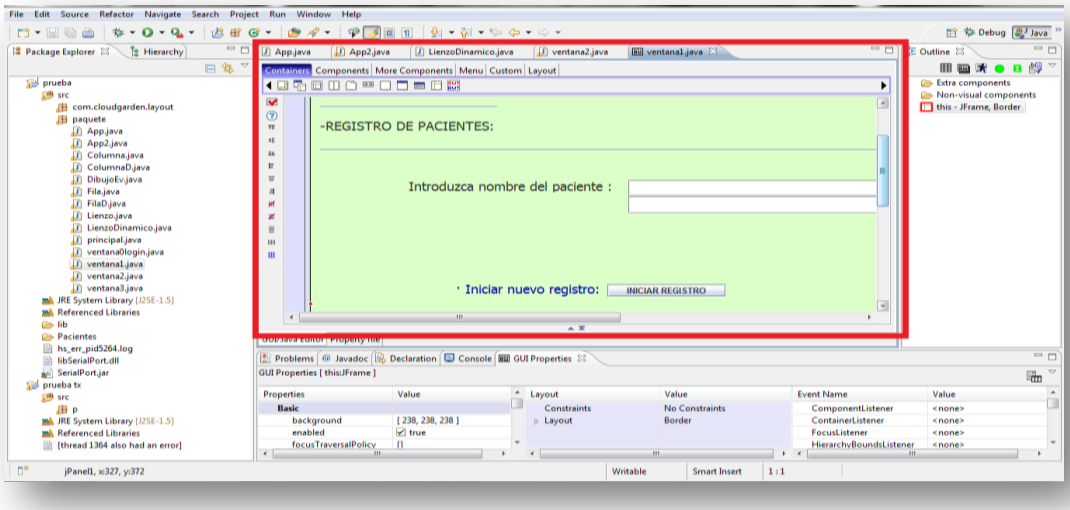



Figura 3.2.2. Funcionalidad añadida por el plugin jigloo, constructor de interfaces gráficas.

## 3.3 DISEÑO GRÁFICO Y FUNCIONAL

Como hemos mencionado anteriormente, la aplicación consta de varias pantallas que se suceden según las opciones de usuario. A continuación explicaremos el diseño gráfico y funcional, cuyo principal objetivo es ser lo más intuitivo posible cubriendo todas las especificaciones de usuario.

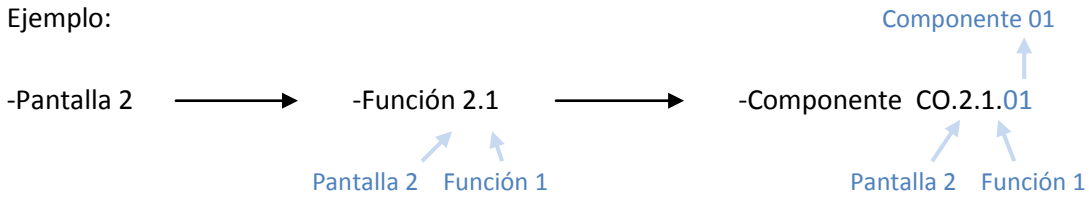
*NOMENCLATURA DE COMPONENTES GRÁFICOS:*

Nombraremos los componentes gráficos con el prefijo "CO".
A continuación escribiremos el numero de pantalla a la que pertenece y seguidamente el número de la función en la que se encuentra involucrado. Por último escribiremos el número de componente.

- Los nombres de componentes quedarán de la siguiente manera:

CO. *Número de pantalla. Número de función. Número de componente*

- Ejemplo:



## PANTALLA 1. PANTALLA DE INICIO

El propósito de esta pantalla de inicio es dar la bienvenida al usuario y proteger el sistema con una contraseña. Una vez introducida la contraseña correctamente da paso a la siguiente pantalla.

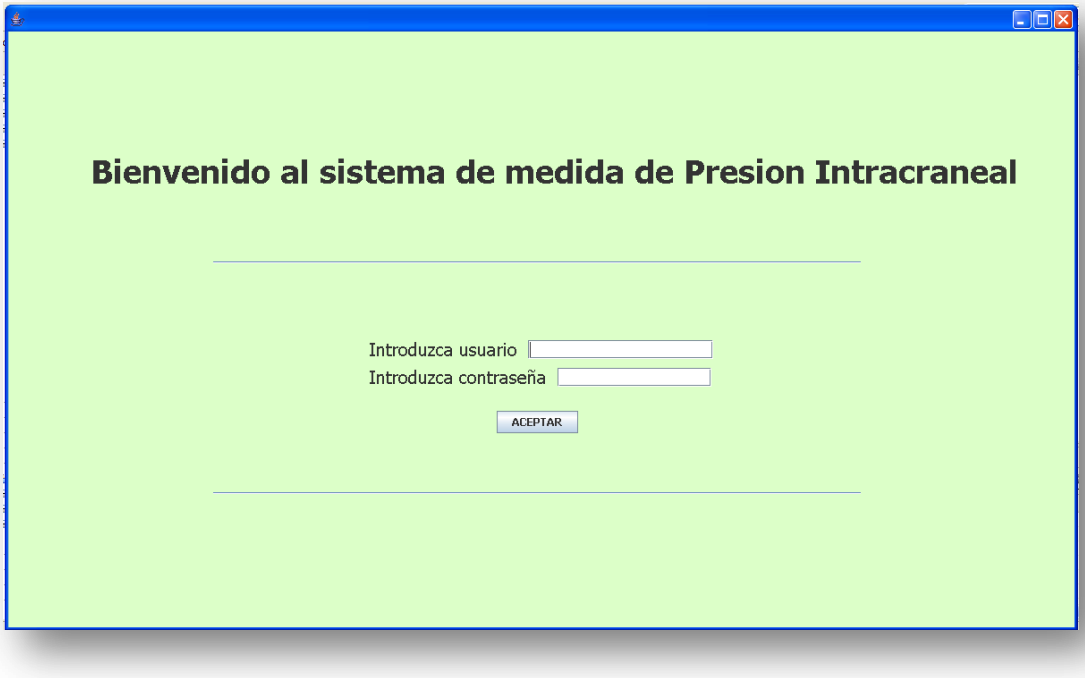



Figura 3.3.1. Pantalla de bienvenida.

### *FUNCIONES DE LA PANTALLA Y COMPONENTES GRÁFICOS UTILIZADOS*

- **Función 1.1 Acceso protegido con contraseña.**

  Para comenzar a utilizar la aplicación es preciso introducir un nombre de usuario y una contraseña, que en este caso serán el mismo para todo el personal cualificado. Hecho esto, presionando el botón "ACEPTAR", y si la contraseña y usuario son correctos, dará paso a la siguiente pantalla.

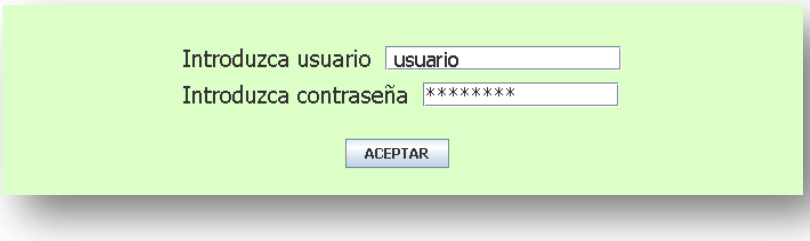
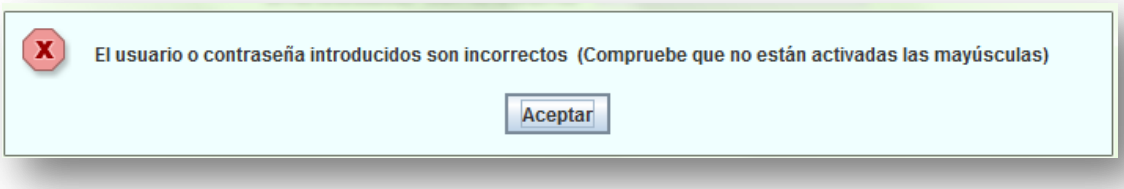



Figura 3.3.2. Función 1.1 Acceso protegido con contraseña

Hay que prestar especial atención a las mayúsculas y minúsculas, ya que Java es sensible a estas, es decir, las reconoce como caracteres diferentes.

*Mensajes de error posibles:*

En el caso de que el usuario o contraseña no sean correctos lanzará el siguiente mensaje de error.




*Componentes gráficos utilizados:*

- CO.1.1.01 Etiqueta: "Bienvenido al Sistema de Medida de Presión Intracraneal"
  Componente Java: JLabel
- CO.1.1.02 Etiqueta: "Introduzca usuario"
  Componente Java: JLabel
- CO.1.1.03 Etiqueta: "Introduzca contraseña"
  Componente Java: JLabel
- CO.1.1.04 Campo de texto (usuario)
  Componente Java: JTextField
- CO.1.1.05 Campo de texto (contraseña)
  Componente Java: JPasswordField
- CO.1.1.06 Botón: "ACEPTAR"
  Componente Java: JButton

## PANTALLA 2. PANTALLA DE REGISTRO Y SELECCIÓN DE PACIENTES.

Esta pantalla tiene dos propósitos generales, y por tanto está dividida en dos áreas.

- La primera permite el registro/selección de pacientes para iniciar una nueva monitorización o ver una antigua.
- La segunda permite la calibración del monitor.

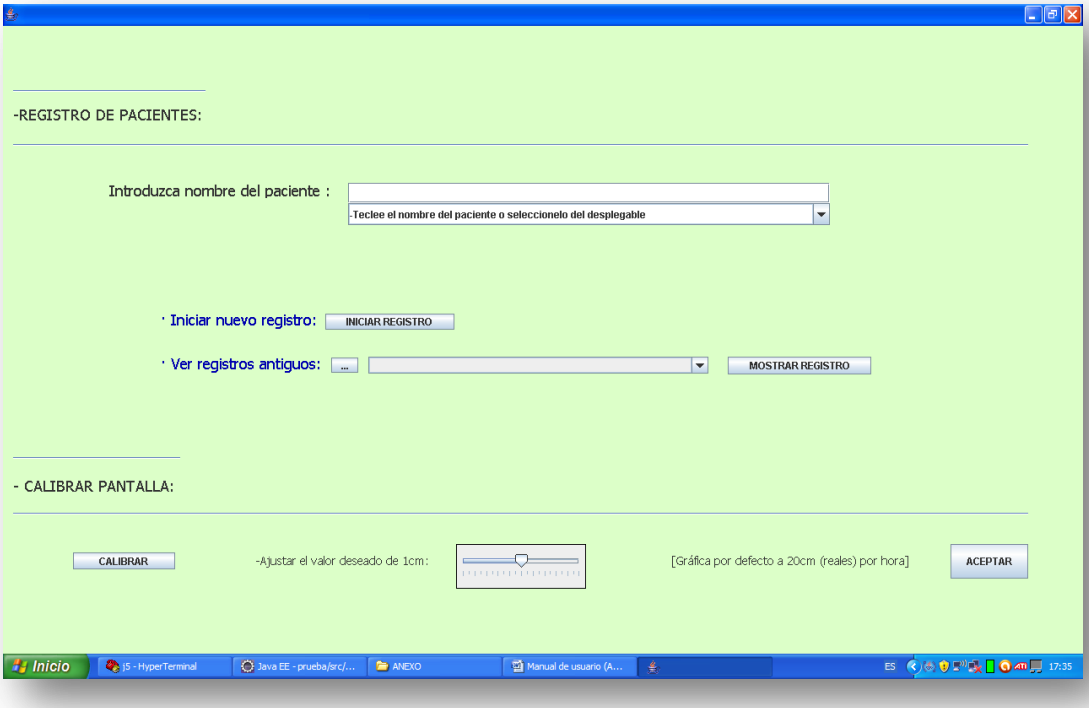
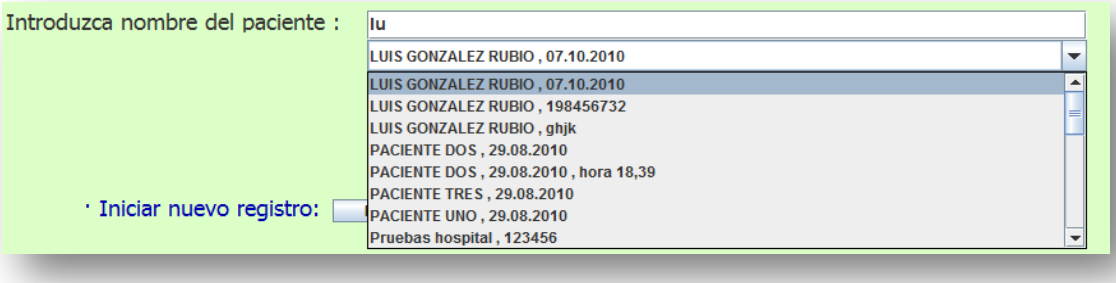



Figura 3.3.3. Pantalla de Registro de Pacientes y Calibración.

### *FUNCIONES DE LA PANTALLA Y COMPONENTES GRÁFICOS UTILIZADOS*

#### Función 2.1 Buscador de pacientes.

Disponemos de un campo de texto junto a la etiqueta de "Introduzca nombre del paciente", según vamos tecleando caracteres en este campo aparecerá debajo un desplegable con una selección de nombres de los pacientes registrados que más se ajustan a lo que buscamos.



Los nombres de los pacientes se presentan de forma que el paciente elegido como más probable aparece en primer lugar, seguido de los correspondientes nombres que le sucederían en orden alfabético.

*Mensajes de error posibles:*

No lanza mensajes de error.

*Componentes gráficos utilizados:*

- CO.2.1.01 Etiqueta: "Introduzca nombre del paciente"
  Componente Java: JLabel
- CO.2.1.02 Campo de texto
  Componente Java: JTextField
- CO.2.1.03 Desplegable de pacientes registrados
  Componente Java: JComboBox

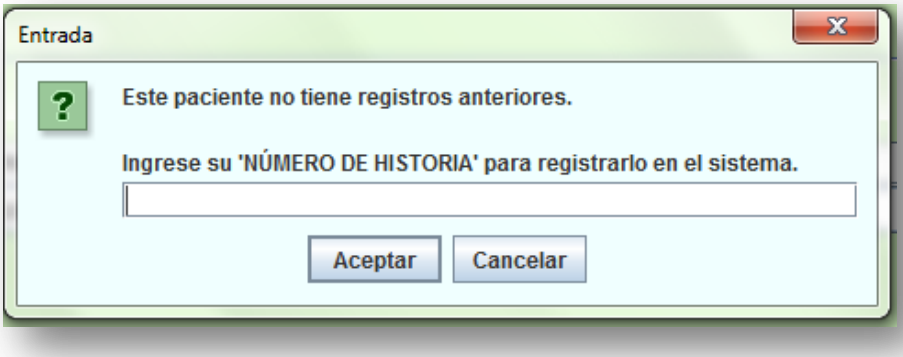
**Función 2.2 Inicio de monitorización.**

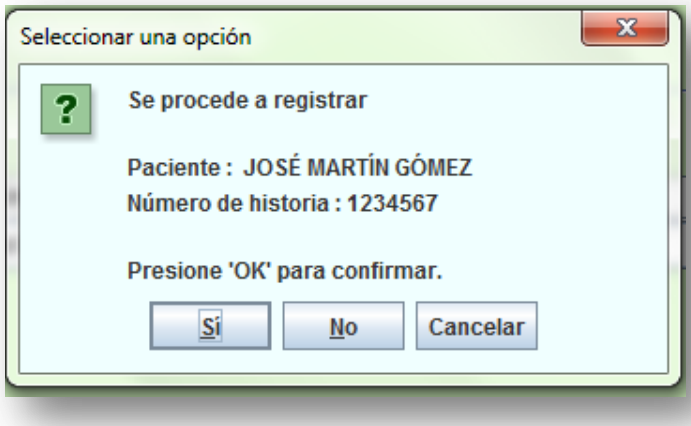
Iniciar la monitorización dará paso a la pantalla siguiente (pantalla 3), dedicada a la monitorización de la PIC actual, así como la creación de un directorio para el paciente (*apdo. 3.2.1. 'Directorio de pacientes'*) si este es de nuevo ingreso.


Figura 3.3.4. Función 2.2 Inicio de monitorización.

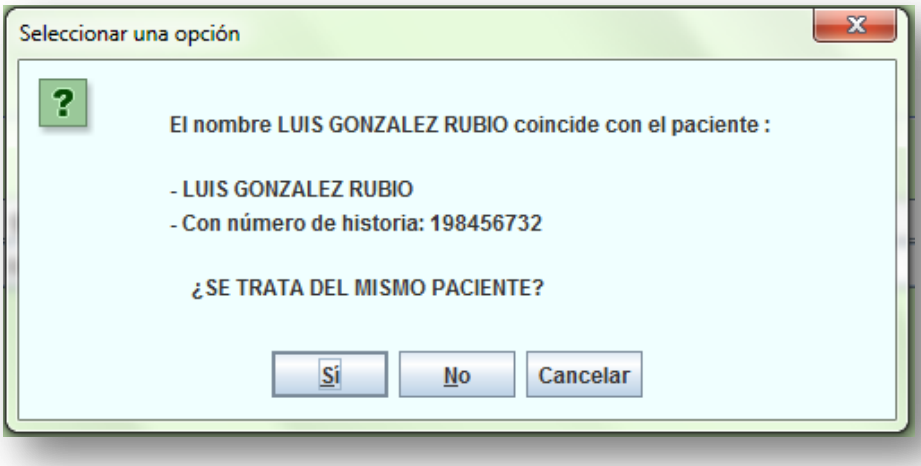
Comprueba si el paciente es de nuevo ingreso o no, y da paso a la pantalla siguiente.

En caso de que el nombre del paciente no aparezca en el directorio se pide introducir por pantalla su número de historia para proceder a su registro.






En caso de que el nombre del paciente tecleado coincida con un nombre de paciente registrado, se pide por pantalla confirmación.




*Mensajes de error posibles:*

No lanza mensajes de error.

*Componentes gráficos utilizados:*

- CO.2.2.01 Etiqueta: "Iniciar nuevo registro"
  Componente Java: JLabel
- CO.2.2.02 Botón: "INICIAR REGISTRO"
  Componente Java: JButton

**Función 2.3 Ver registros antiguos.**

Esta opción sirve para ver un registro anterior del paciente seleccionado, esto permite a los médicos ver detalles de otras gráficas de PIC y seguir la evolución del paciente.

Una vez seleccionado el registro, éste se abre en la pantalla 4, pantalla de estudio de la curva.

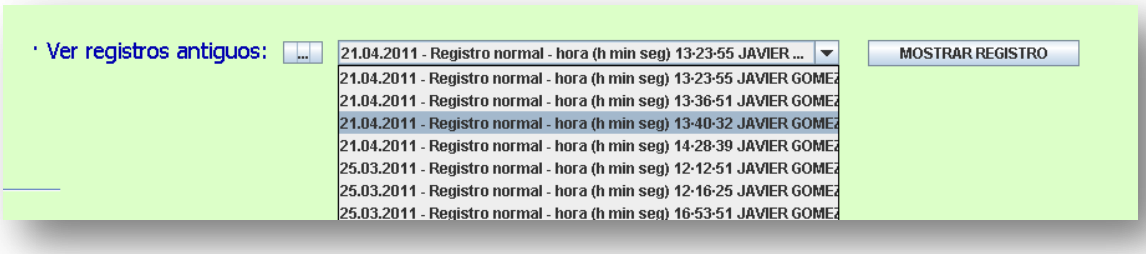
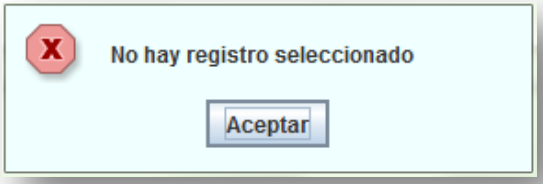


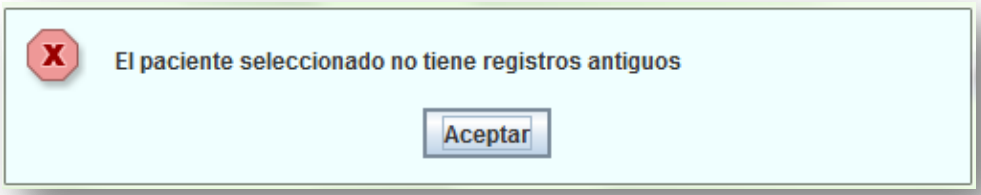

Figura 3.3.5. Ver Registros antiguos

*Mensajes de error posibles:*

Si no hay registro seleccionado se mostrará el siguiente mensaje:




Si el paciente seleccionado no tiene registros antiguos se mostrará este otro:




*Componentes gráficos utilizados:*

- CO.2.3.01 Etiqueta: "Ver registros antiguos"
  Componente Java: JLabel
- CO.2.3.02 Botón: "..."
  Componente Java: JButton
- CO.2.3.03 Desplegable: registros antiguos del paciente seleccionado
  Componente Java: JComboBox
- CO.2.3.04 Botón: "MOSTRAR REGISTRO"
  Componente Java: JButton

**Función 2.4 Calibrar pantalla.**

Como ya hemos mencionado anteriormente la gráfica de la PIC se representa por defecto a 20 cm/hora, pero en ocasiones se dispone de monitores más pequeños y para que este parámetro permanezca inalterable hay que ajustar el valor de un centímetro real.

Para este propósito tenemos esta función. Su uso es sencillo, con una regla de medida sobre el monitor, y en concreto sobre esta barra de desplazamiento, se ajustará la medida de un

centímetro real. De esta manera sea cual sea el tamaño del monitor siempre se podrá ajustar la gráfica de la PIC a 20 cm reales/hora

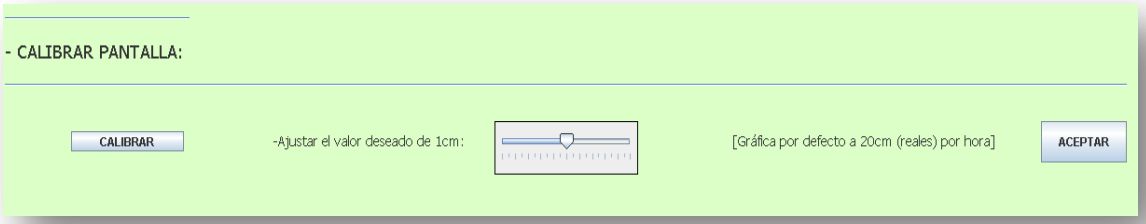



Figura 3.3.6. Función 2.4 Calibrar pantalla.

*Mensajes de error posibles:*

No lanza mensajes de error

*Componentes gráficos utilizados:*

- CO.2.4.01 Etiqueta: "CALIBRAR PANTALLA"
  Componente Java: JLabel
- CO.2.4.02 Botón: "…"
  Componente Java: JButton
- CO.2.4.03 Etiqueta: "Ajustar el valor deseado de un cm"
  Componente Java: JLabel
- CO.2.4.04 Barra de desplazamiento para elegir el valor de un cm
  Componente Java:
- CO.2.4.05 Etiqueta: "[Gráfica por defecto a 20 cm (reales por hora)]"
  Componente Java: JLabel
- CO.2.4.06 Botón: "ACEPTAR"
  Componente Java: JButton

## PANTALLA 3. PANTALLA DE MONITORIZACIÓN DE LA PIC

Esta es la pantalla donde se representan los datos de la PIC extraídos del sensor en tiempo real. El propósito principal de esta pantalla es, por tanto, mostrar la curva de PIC, además de permitir otras funciones como el registro manual de eventos o el acceso a registros antiguos.

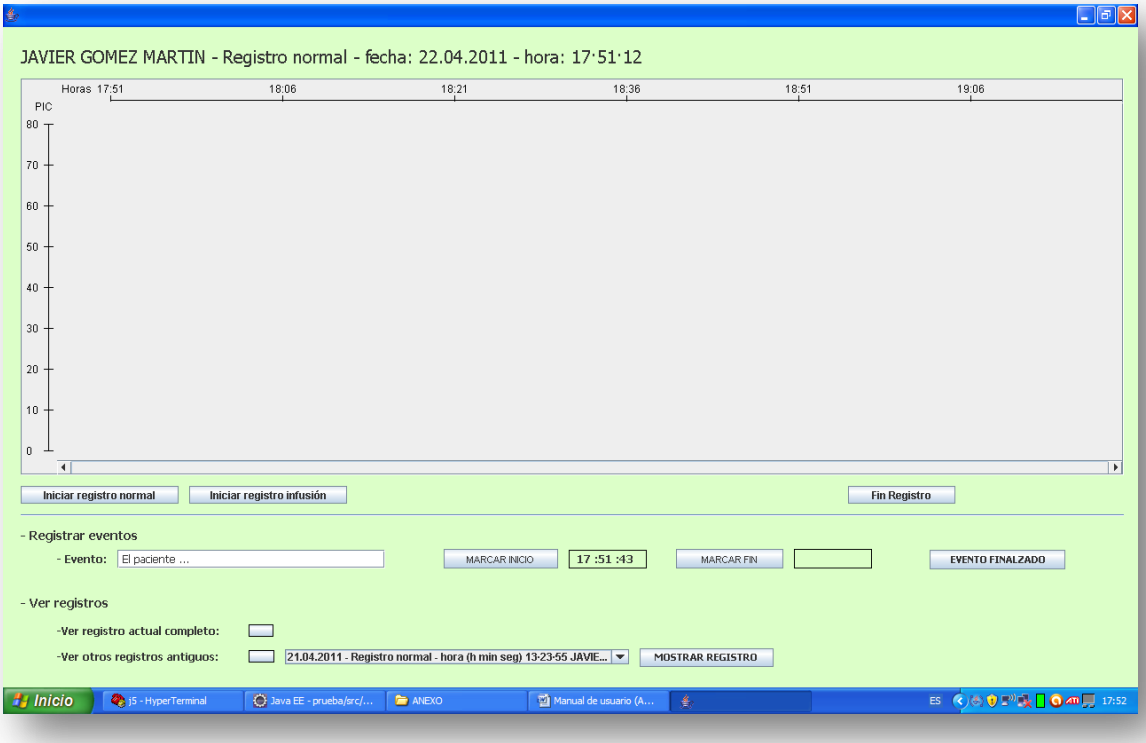



Figura 3.3.7. Pantalla de monitorización de la PIC.

### *FUNCIONES DE LA PANTALLA Y COMPONENTES GRÁFICOS UTILIZADOS*

#### Función 3.1 Monitorización de la PIC

Para la monitorización de la PIC disponemos de un espacio (Lienzo) donde se representa la curva de los valores de la PIC que obtiene el sensor, y que se reciben a través de una conexión serie RS232.

Los valores se representan a medida que se van recibiendo, en concreto esta pantalla se va actualizando cada 8 segundos.

PROCEDIMIENTO:

1. Crear los ficheros correspondientes para esta nueva monitorización:
   - Fichero de PIC.
   - Fichero de eventos.

2. Establecer la comunicación con el monitor de PIC, es decir, determinar los parámetros de la comunicación, abrir el puerto y comenzar a recibir datos.
3. Para cada dato recibido:

    - Evaluar si ha sido recibido durante un evento (ya que estos tramos se pintan de diferente color)
    - Escribirlo junto con la hora actual y un 'indicador de evento' ( 0=no hay evento, 1=hay evento) en el fichero de PIC para posteriores consultas.

```
18/06/2011 , 04:40:21 , 3.1 , 0
18/06/2011 , 04:40:23 , 3.1 , 0
18/06/2011 , 04:40:24 , 3.1 , 0
18/06/2011 , 04:40:25 , 3.1 , 0
18/06/2011 , 04:40:26 , 3.1 , 0
18/06/2011 , 04:40:28 , 3.1 , 0
```

    - Meterlo en una matriz, en la última posición, para representarlo por pantalla.

4. Cada 8 valores recibidos (8 segundos) se actualiza la pantalla con los nuevos datos, es decir se produce un repintado del componente Lienzo.
5. Fin de la monitorización, se cierra la conexión con el monitor de PIC, y los ficheros donde se están escribiendo los datos

TRAMOS DE EVENTOS

Los valores recibidos durante un evento aparecerán en azul mientras que los demás se dibujarán en negro.

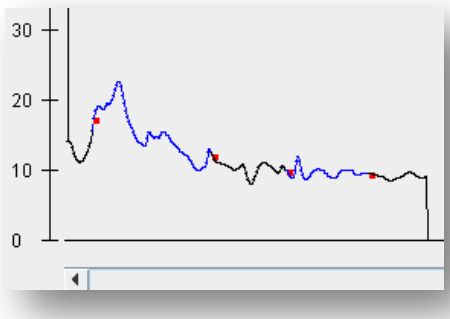



Figura 3.3.8. Dibujo de eventos.

Como podemos ver en la imagen, el comienzo y fin de un evento se marca con un rectángulo rojo, para que se distingan a primera vista estos tramos especiales.

TOOLTIPS

Se han añadido ToolTips en toda la pantalla gráfica (lienzo), los ToolTips son etiquetas que aparecen en un componente cuando dejamos el ratón parado sobre él.

Normalmente el texto del ToolTip es el mismo para todo el componente, que en este caso sería el lienzo, pero eso no es de utilidad para mostrar el valor de la PIC en cada punto.
Para mostrar etiquetas diferentes dentro del mismo componente, cogemos el valor del pixel en que este situado el ratón y se le aplica la siguiente transformación:

*Etiqueta_a_mostrar= (nºpixeles de alto de la pantalla -posicionRaton)/4.6*

Esto permite situar el ratón en cualquier punto de la curva y saber con exactitud a que valor de PIC corresponde.

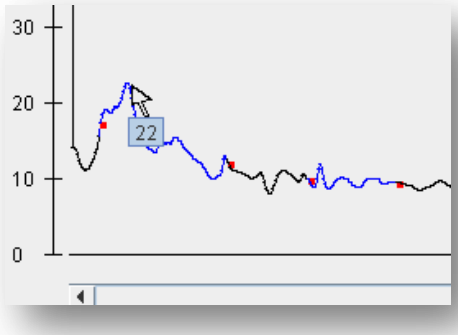



Figura 3.3.9. ToolTips en el lienzo.

### *Mensajes de error posibles:*

No lanza mensajes de error.

### *Componentes gráficos utilizados*

- CO.3.1.01 Lienzo donde se representan los valores en tiempo real de la PIC.
  Componente Java: Componente implementado (Lienzo).
  Implementar un componente implica dibujarlo a mano, en vez de coger uno de la librería de componentes gráficos predeterminados de Java.
  En esta ocasión es un componente en blanco, hasta que comienzan a llegar valores de PIC, y podemos empezar a representarlos.

  El eje de coordenadas en un componente gráfico en java está situado en la esquina superior izquierda. El componente "Lienzo" tiene 400 píxeles de altura, y por tanto para representar los valores de PIC acorde a el eje vertical (CO.3.1.02 ) tenemos que hacer una transformación sobre estos:

  *Pixel_en _el_que_dibujo = nºpixeles de alto del lienzo-(valorPICrecibido*4.6)*

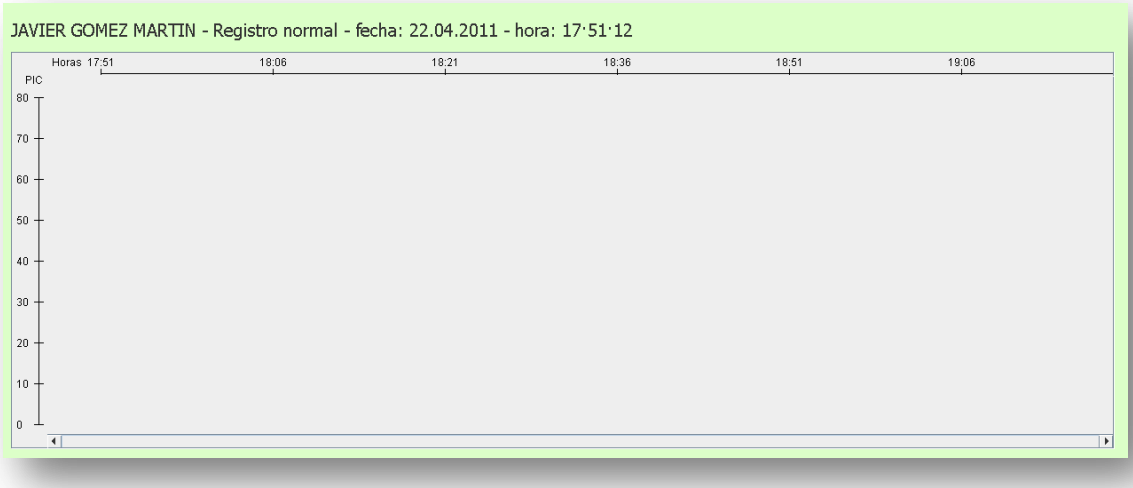


Figura 3.3.10.Componente “Lienzo” donde se representa la PIC.

- CO.3.1.02 Eje vertical, escala de PIC
  Componente Java: Componente implementado (Eje vertical).
  Implementar un componente implica dibujarlo a mano, en vez de coger uno de la librería de componentes gráficos predeterminados de Java.

  El eje vertical es una escala que se coloca junto al Lienzo, y marca el valor de la PIC.

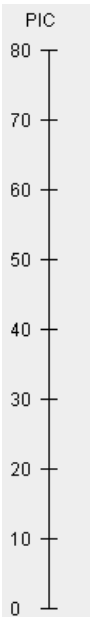



Figura 3.3.11. Eje vertical.

- CO.3.1.03 Eje horizontal, escala temporal
  Componente Java: Componente implementado (Eje horizontal).
  Implementar un componente implica dibujarlo a mano, en vez de coger uno de la librería de componentes gráficos predeterminados de Java.

  El eje horizontal es una escala que se coloca encima del Lienzo y marca el transcurso del tiempo. El primer valor es la hora de comienzo del registro, y los intervalos de tiempo se marcan cada 15 minutos en el eje, ocupando 20 centímetros por cada hora de monitorización.

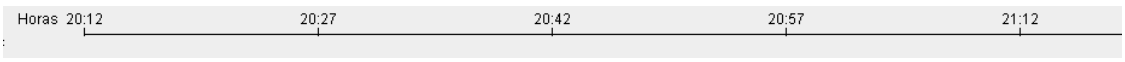



Figura 3.3.12. Eje horizontal del lienzo.

- CO.3.1.04 Contenedor del lienzo y los ejes, con barras de desplazamiento para la gráfica.
  Componente Java: JscrollPane
  La aplicación traza 20 cm de lienzo por hora de muestreo.
  Las monitorizaciones de la PIC pueden durar varias horas, y llega un momento en que el lienzo debe aumentar su tamaño para contener todos los datos.

  Este componente contiene el lienzo, aportando barras de desplazamiento, tanto verticales como horizontales para ver en su totalidad el registro de PIC.

  También contiene los ejes ('Eje vertical ' y 'Eje horizontal') mostrándolos siempre visibles y en concordancia con el desplazamiento de la barra.

**Función 3.2 Registros normales separados de registros de infusión**

Se debe diferenciar si la monitorización es un registro normal o de infusión. Para ello se han creado dos botones, uno para iniciar un registro normal y otro para registros de infusión. Esto nombrará los ficheros acorde al tipo de monitorización que se esté realizando. También se mostrará en el encabezado de la pantalla el nombre del paciente junto al tipo de monitorización.

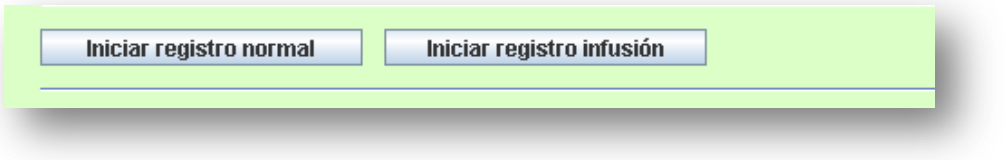



Figura 3.3.13. Función 3.2 Registros normales separados de registros de infusión.

*Mensajes de error:*

No lanza mensajes de error

*Componentes gráficos utilizados:*

- CO.3.2.01 Botón: "Iniciar registro normal"
  Componente Java: JButton
- CO.3.2.02 Botón: "Iniciar registro infusión"
  Componente Java: JButton

**Función 3.3 Marcar manualmente eventos.**

Es necesario poder marcar si está ocurriendo algún evento durante la monitorización, ya que los valores de PIC llegados en ese intervalo de tiempo son valores que hay que descartar cuando se hace un análisis.
Como hemos dicho anteriormente, estos tramos de tiempo especiales se representarán en un color distinto a los tramos con valores ordinarios.

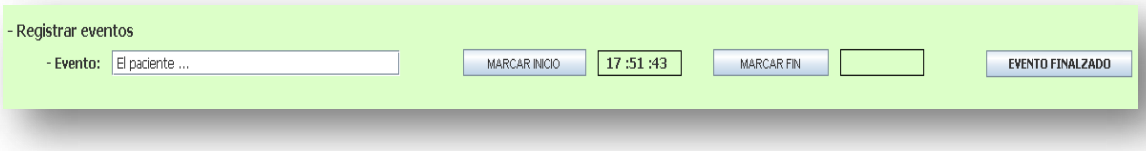



Figura 3.3.14. Función 3.3 Marcar manualmente eventos.

Se escribirá el nombre del evento, se marcará la hora de comienzo y fin, y por último, pulsando el botón "Evento finalizado" se borrarán todos los campos para poder registrar un nuevo evento. Los datos de los eventos aquí registrados se guardarán en la carpeta de eventos correspondientes a dicha monitorización.

*Mensajes de error posibles:*

No lanza mensajes de error.

*Componentes gráficos utilizados:*

- CO.3.3.01 Etiqueta: "Registrar eventos"
  Componente Java: JLabel
- CO.3.3.02 Etiqueta: " -Evento: "
  Componente Java: JLabel
- CO.3.3.03 Campo de texto para escribir el evento
  Componente Java: JTextField
- CO.3.3.04 Botón: "MARCAR INICIO"
  Componente Java: JButton
- CO.3.3.05 Etiqueta: donde se marca el inicio
  Componente Java: JLabel
- CO.3.3.06 Botón: "MARCAR FIN"
  Componente Java: JButton
- CO.3.3.07 Etiqueta: donde se marca el fin
  Componente Java: JLabel
- CO.3.3.08 Botón: "COMPLETADO"
  Componente Java: JButton

### Función 3.4 Finalizar monitorización

Al pulsar este botón se finaliza el registro actual, es decir, se cierra la conexión con el puerto serie y se cierran los ficheros en los que se estaban escribiendo los datos recibidos.

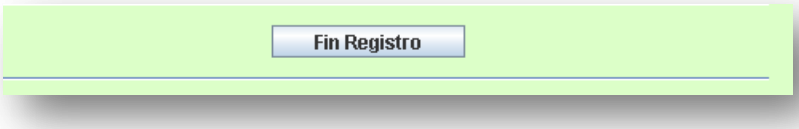



Figura 3.3.15. Función 3.4 Finalizar monitorización.

*Mensajes de error posibles:*

No lanza mensajes de error.

*Componentes gráficos utilizados:*

- CO.3.4.01 Botón: “Fin registro”
  Componente Java: JButton

**Función 3.5 Acceder a la pantalla de estudio de la curva para el registro actual**

Abre el registro actual en la ventana de estudio de la curva. Esto es útil para ver en detalle la curva de la PIC que se está monitorizando, y aplicar sobre ella las opciones que la pantalla 4 ofrece (aumentar, disminuir, marcar basal,…).

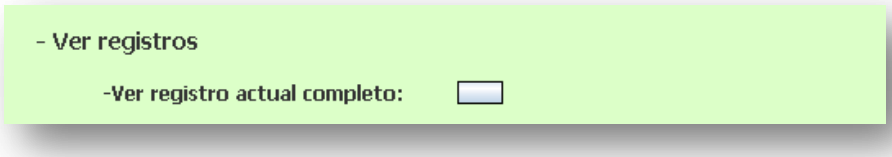



Figura 3.3.16. Acceder a la pantalla de estudio de la curva para el registro actual.

*Mensajes de error posibles:*

No lanza mensajes de error.

*Componentes gráficos utilizados:*

- CO.3.5.01 Etiqueta: “ -Ver registros ”
  Componente Java: JLabel
- CO.3.5.02 Etiqueta: “ -Ver registro actual completo ”
  Componente Java: JLabel
- CO.3.5.03 Botón acceso
  Componente Java: JButton

**Función 3.6 Acceder a la pantalla de estudio de la curva para un registro antiguo**

Es interesante poder ver otros registros del paciente, esto permite ver su evolución y detectar sus patrones de ondas.

Una vez elegido el registro antiguo de un desplegable, se abrirá en la ventana de estudio de la curva (pantalla 4).

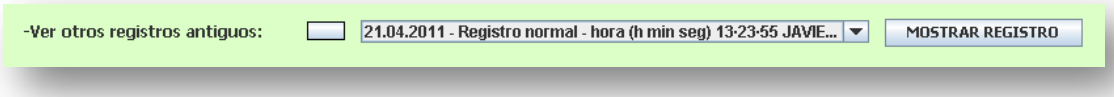
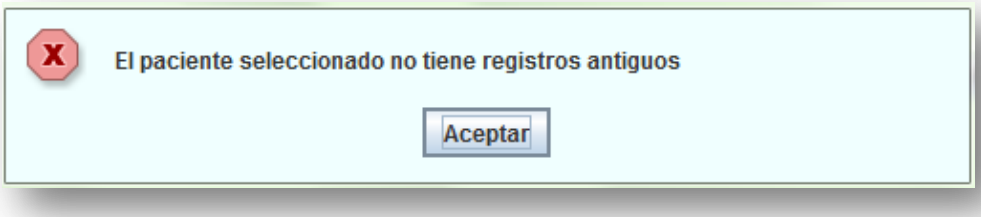



Figura 3.3.17. Función 3.6 Acceder a la pantalla de estudio de la curva para un registro antiguo.


Figura 3.3.18.Desplegable ver registros antiguos.

*Mensajes de error posibles:*

Si el paciente actual no tiene registros antiguos se mostrará el siguiente mensaje




*Componentes gráficos utilizados:*

- CO.3.6.01 Etiqueta: " –Ver otros registros antiguos "
  Componente Java: JLabel
- CO.3.6.02 Botón acceso
  Componente Java: JButton
- CO.3.6.03 Desplegable: registros antiguos del paciente
  Componente Java: JComboBox
- CO.3.6.04 Botón: "MOSTRAR REGISTRO"
  Componente Java: JButton

## PANTALLA 4. PANTALLA DE ESTUDIO DE LA CURVA.

El propósito de esta pantalla es mostrar la curva de la PIC (actual o de una monitorización antigua) ofreciendo diversas funciones que permiten su estudio.

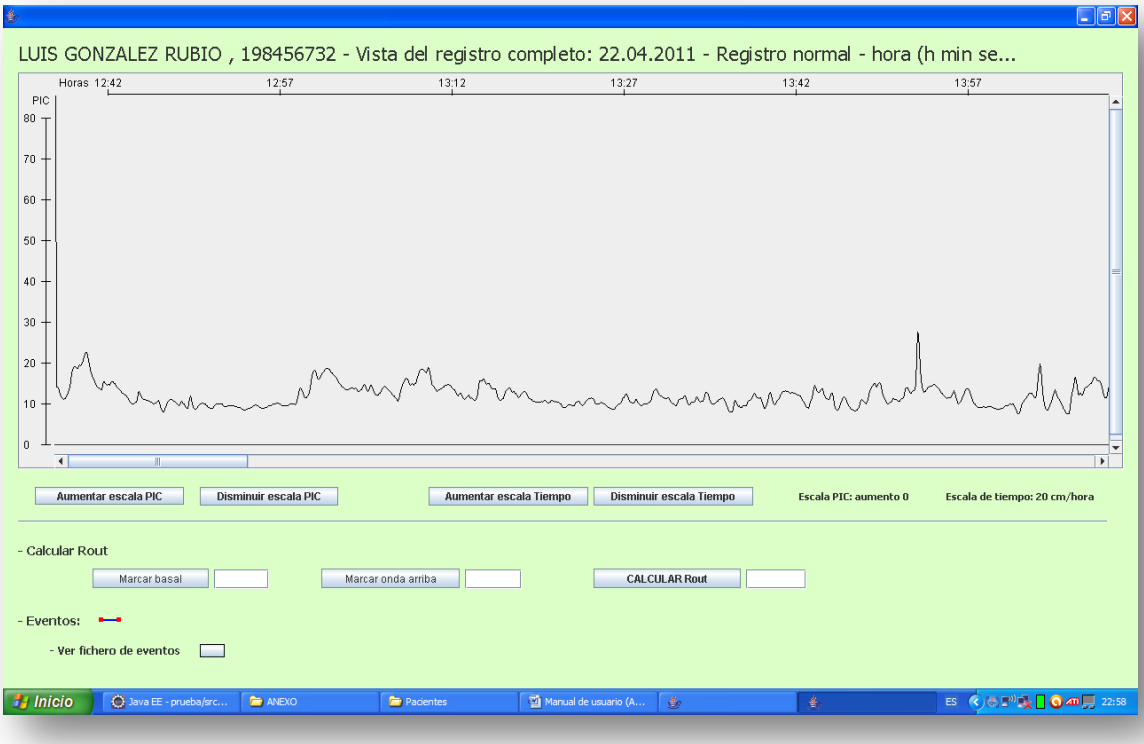



Figura 3.3.19.Pantalla de estudio de la curva.

### *FUNCIONES DE LA PANTALLA Y COMPONENTES GRÁFICOS UTILIZADO*

#### Función 4.1 Mostrar el registro seleccionado.


Figura 3.3.20. Función 4.1 Mostrar el registro seleccionado.

Es la función principal de la pantalla, consiste en representar la curva de la PIC en un componente gráfico habilitado para ello (“Lienzo”).

TRAMOS DE EVENTOS
Los valores recibidos durante un evento aparecerán en azul mientras que los demás se dibujarán en negro.

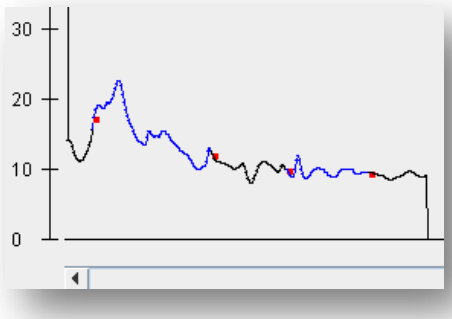



Figura 3.3.21. Dibujo de eventos.

Como podemos ver en la imagen, el comienzo y fin de un evento se marca con un rectángulo rojo, para que se distingan a primera vista estos tramos especiales.

TOOLTIPS
Se han añadido ToolTips en toda la pantalla gráfica (lienzo), los ToolTips son etiquetas que aparecen en un componente cuando dejamos el ratón parado sobre él.
Esto permite situar el ratón en cualquier punto de la curva y saber con exactitud a que valor de PIC corresponde.

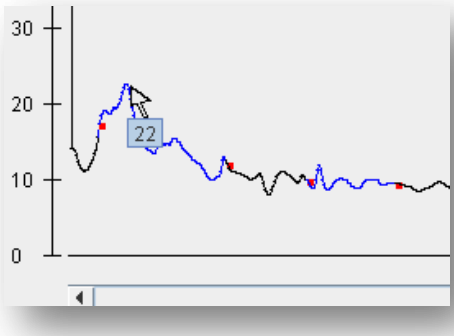



Figura 3.3.22.ToolTips en el lienzo.

*Mensajes de error posibles:*
No lanza mensajes de error.

*Componentes gráficos utilizados:*

- CO.4.1.01 Lienzo donde se representan los valores de un fichero de PIC.
  Componente Java: Componente implementado (véase CO.3.1.01 , Lienzo)
- CO.4.1.02 Eje vertical, escala de PIC
  Componente Java: Componente implementado (véase CO.3.1.02 , Eje vertical)
- CO.4.1.03 Eje horizontal, escala temporal

Componente Java: Componente implementado (véase CO.3.1.02 , Eje horizontal)

- CO.4.1.04 Contenedor para Lienzo con barras de desplazamiento
  Componente Java: JScrollPane

### Función 4.2 Aumentar/Disminuir la escala de la PIC

Aumentar o disminuir el detalle de la escala de PIC es fundamental para detectar patrones de ondas, que a escala natural (escala de la pantalla de monitorización) no se distinguen bien.

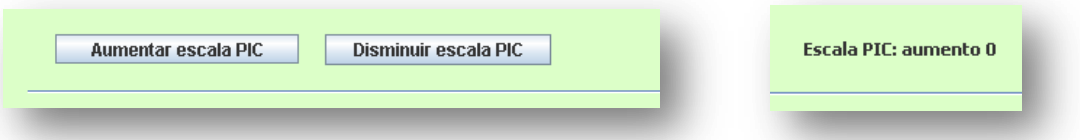
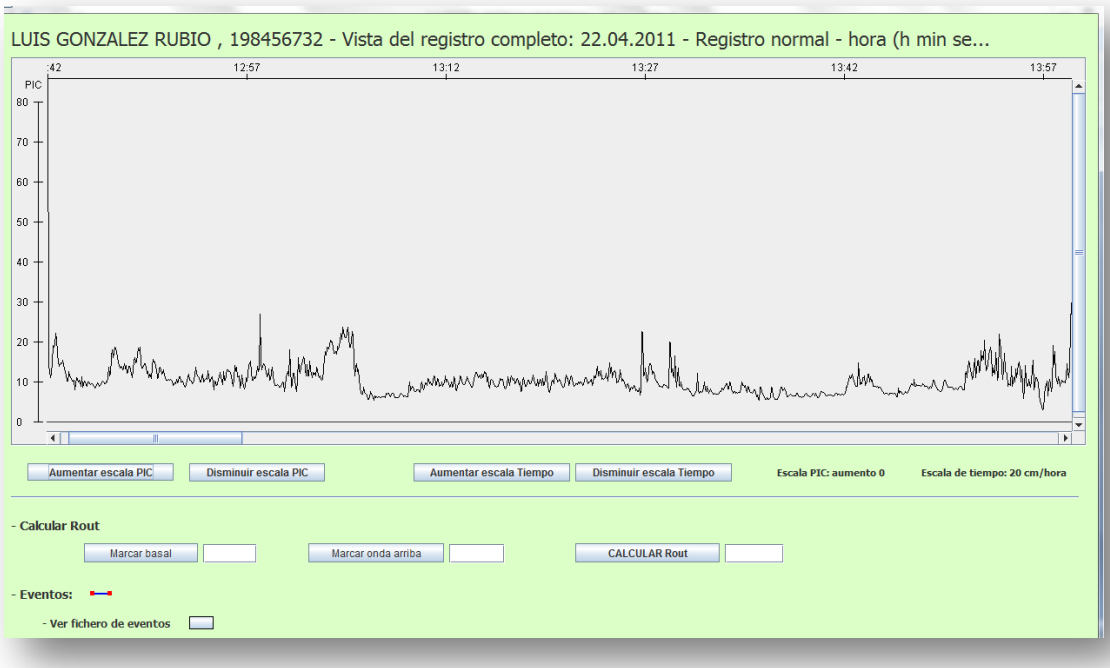


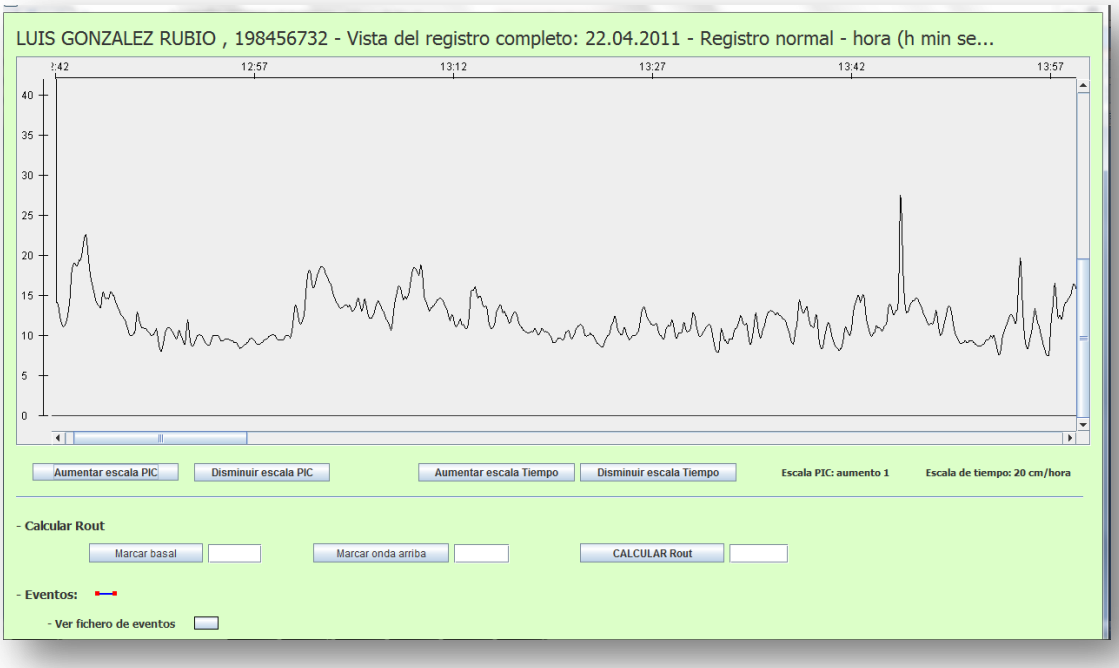

Figura 3.3.23. Aumentar/Disminuir la escala de la PIC

La gráfica inicial, es decir, un lienzo donde se pueden ver simultáneamente todos los valores de la escala de la PIC (de 0 a 80) en la pantalla, es lo más reducido que se puede ver la curva.

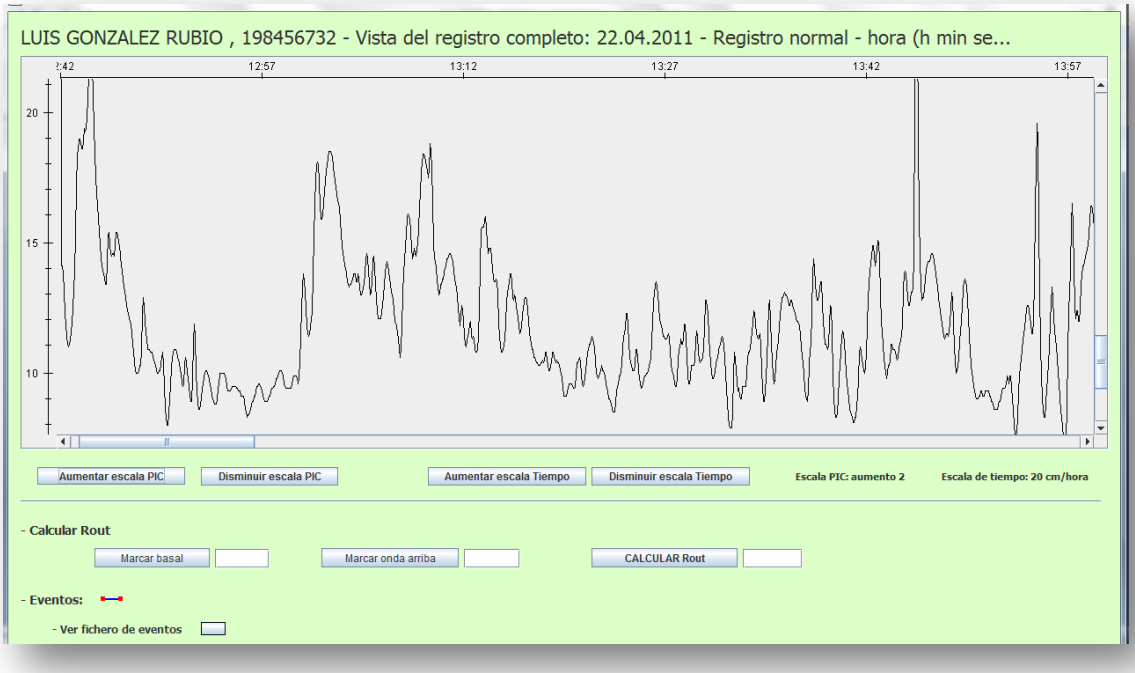



Partiendo de ahí, hay dos niveles más de aumento.
Si estamos en el nivel máximo de aumento y aún así se pulsa el botón otra vez, el nivel de aumento no subirá sino que se mantendrá en esa posición.

Aumento 1:




Aumento 2:




*Mensajes de error posibles:*

No lanza mensajes de error.

*Componentes gráficos utilizados:*

- CO.4.2.01 Botón: “Aumentar escala PIC”
  Componente Java: JButton
- CO.4.2.02 Botón: “Disminuir escala PIC”

Componente Java: JButton

- CO.4.2.03 Etiqueta: " Nivel de aumento: X "
  Componente Java: JLabel

### Función 4.3 Aumentar/Disminuir la escala temporal

Aumentar o disminuir la escala temporal puede ser fundamental para detectar patrones que a escala natural (20 cm/hora) no se distinguen bien.

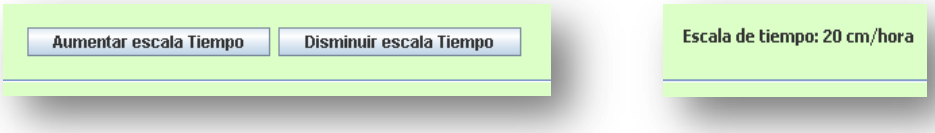
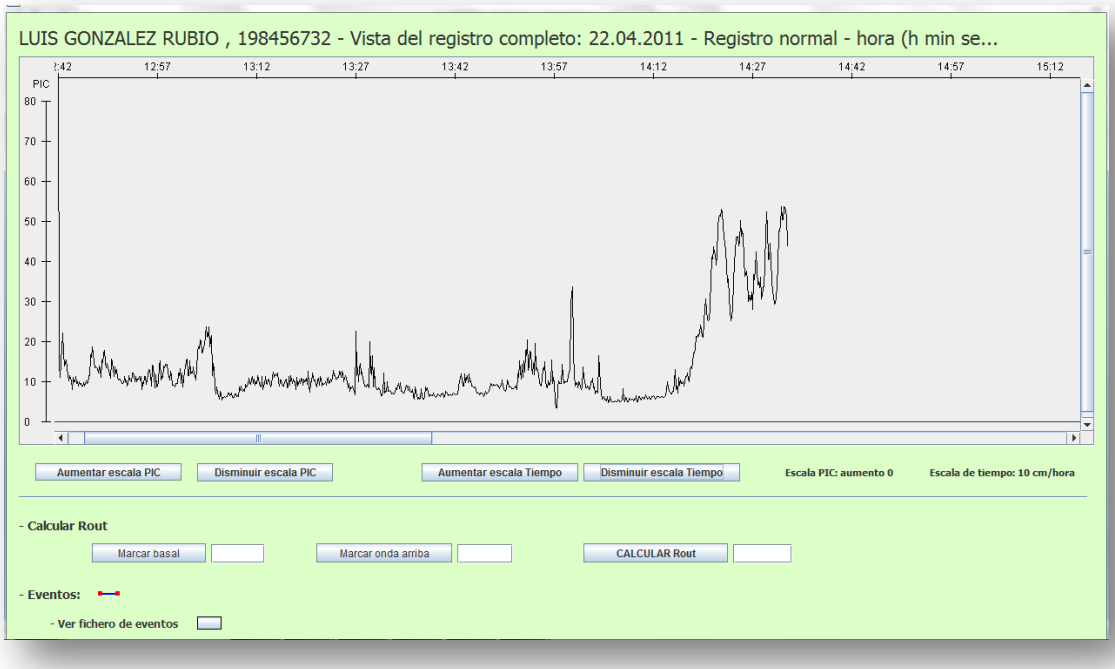



Figura 3.3.24. Aumentar/Disminuir la escala temporal

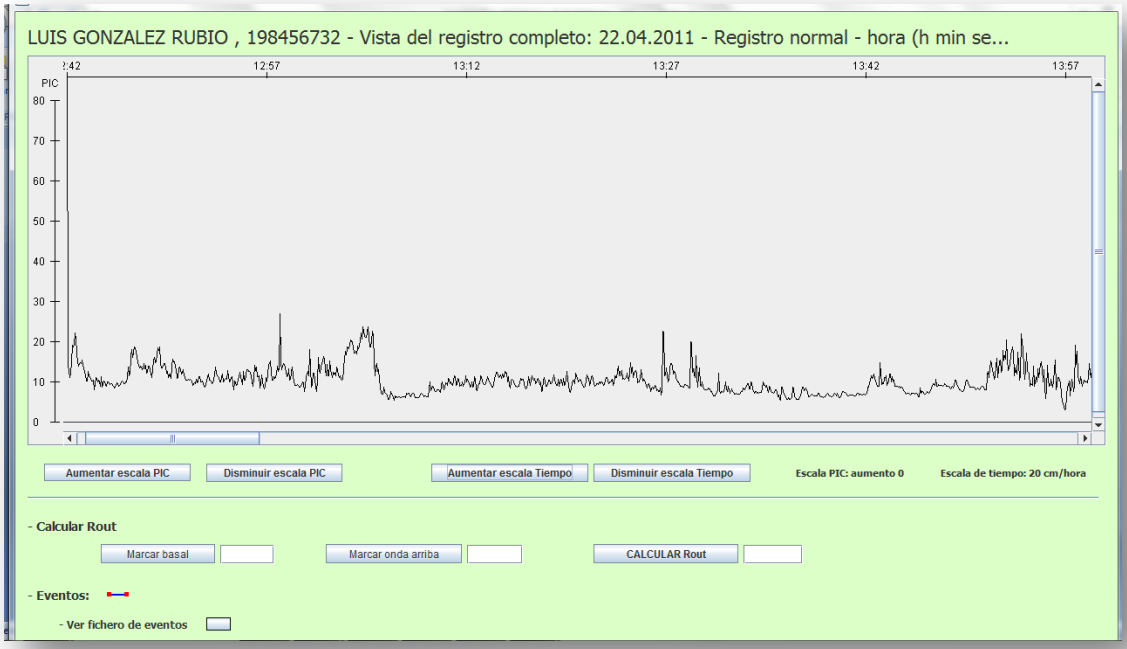
La gráfica inicial representa cada hora de muestreo en 20 cm, es decir a 20 cm/hora, pero esta escala se puede modificar.

Las escalas posibles son todas aquellas que pueden ser de utilidad para el estudio de la curva, y han sido determinadas por los especialistas.

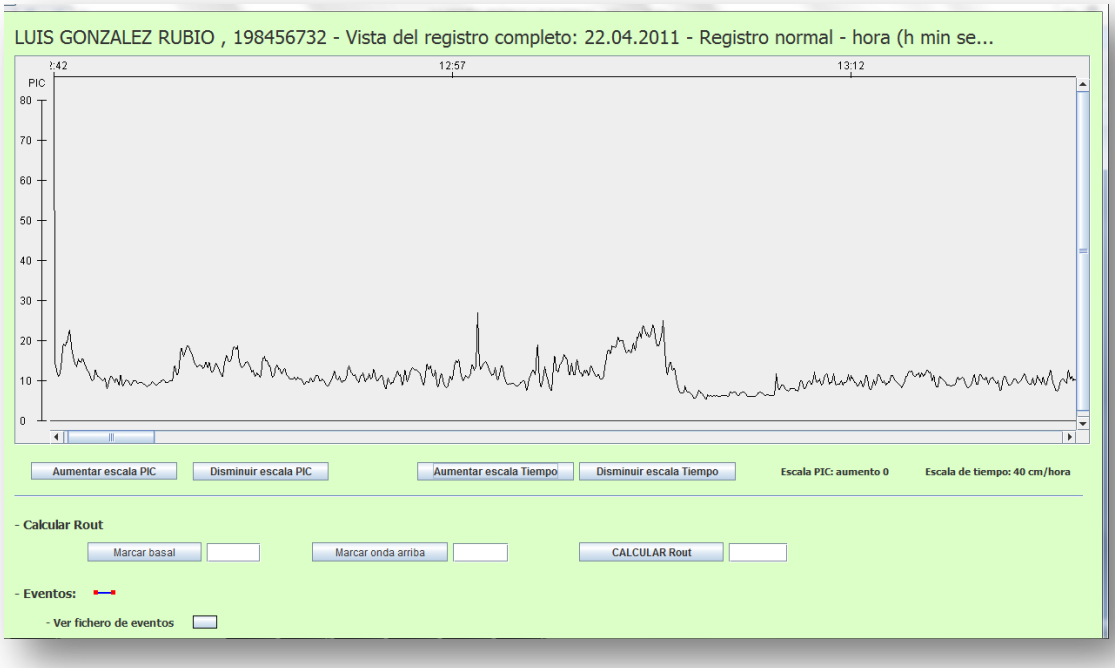
Escala temporal: 10cm/hora



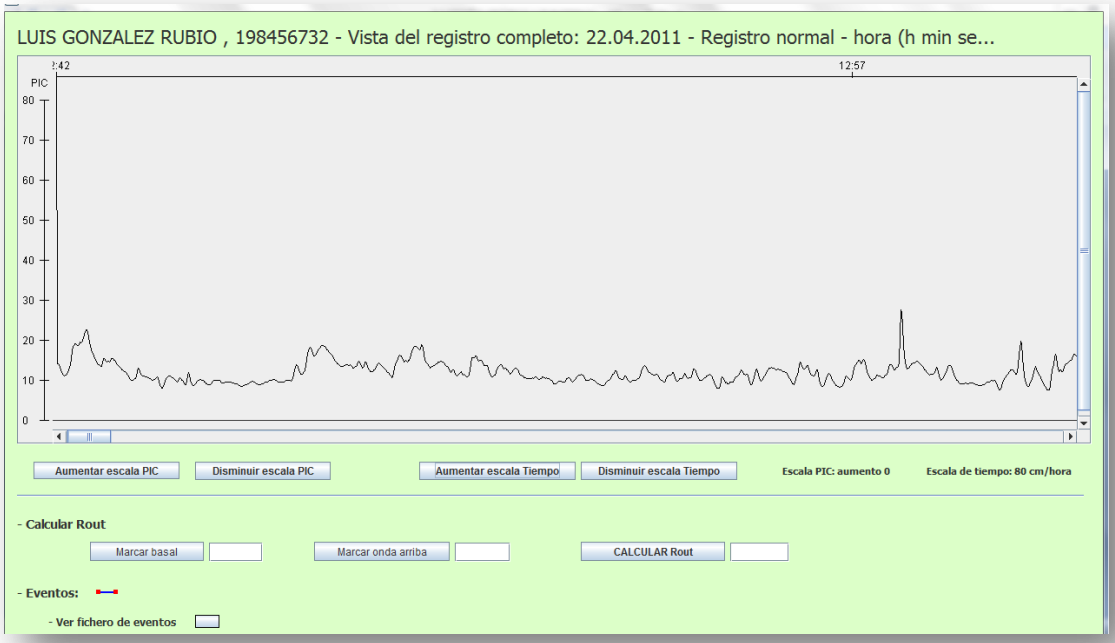
Escala temporal:  20cm/hora




Escala temporal:  40cm/hora



Escala temporal:  80cm/hora




Si estamos en el nivel máximo de escala y aún así se pulsa el botón otra vez, la escala no aumentará sino que se mantendrá en esa posición.

*Mensajes de error posibles:*

No lanza mensajes de error.

*Componentes gráficos utilizados:*

- CO.4.3.01  Botón: “Aumentar escala temporal”
  Componente Java:  JButton
- CO.4.3.02  Botón: “Disminuir escala temporal”
  Componente Java:  JButton
- CO.4.3.03  Etiqueta:  “ Escala temporal:  X ”
  Componente Java:  JLabel

## Función 4.4  Marcar basal/ onda arriba

Estos botones permiten marcar el valor de la basal u onda arriba directamente sobre la pantalla de la gráfica. El valor aparecerá en el campo de texto situado junto a estos.

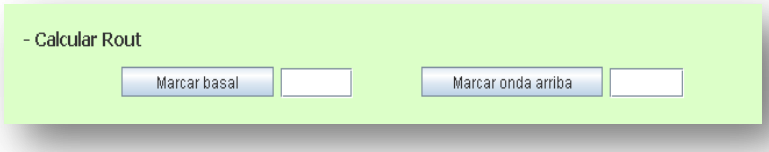
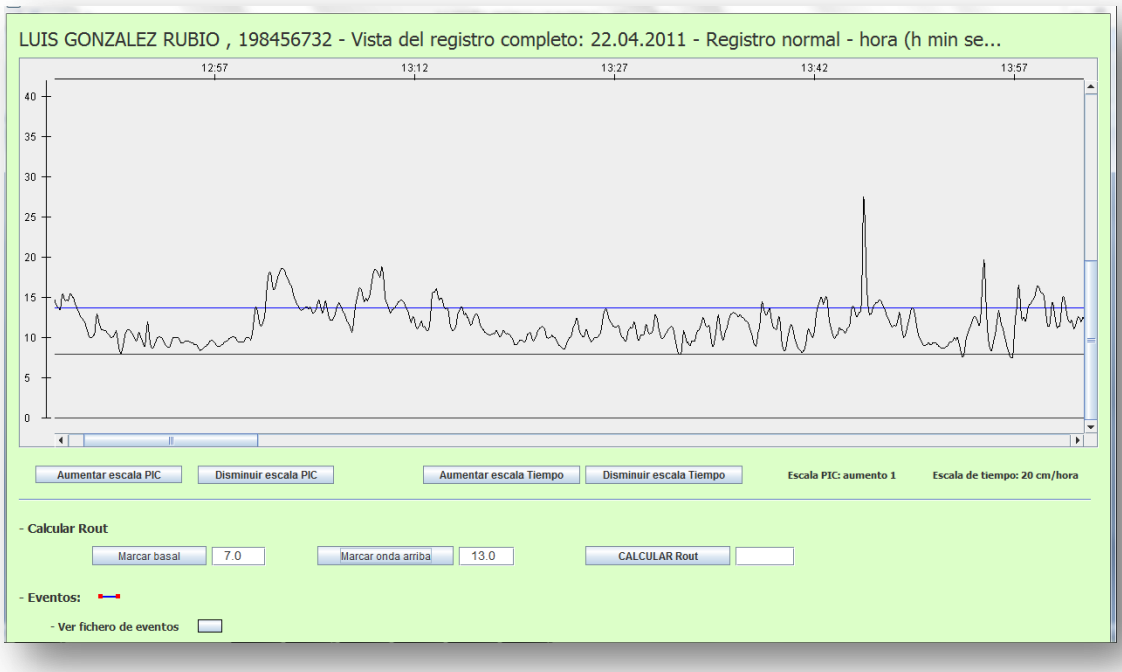


Figura 3.3.25. Función 4.4  Marcar basal/ onda arriba.




*Mensajes de error posibles:*

No lanza mensajes de error.

*Componentes gráficos utilizados:*

- CO.4.4.01  Etiqueta:  " Calcular R.out "
  Componente Java:  JLabel
- CO.4.4.02  Botón: "Marcar basal"
  Componente Java:  JButton

- CO.4.4.03  Campo de texto (marcar basal
  Componente Java:  JTextField
- CO.4.4.04  Botón: "Marcar onda arriba"
  Componente Java:  JButton
- CO.4.4.05  Campo de texto (marcar onda arriba)
  Componente Java:  JTextField

**Función 4.5  Calcular R.out**

Una vez marcados los valores de basal y onda arriba, podremos calcular la R.out pulsando este botón.

El valor aparecerá en el campo de texto situado junto al botón.

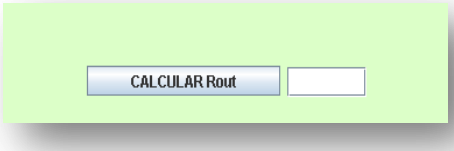
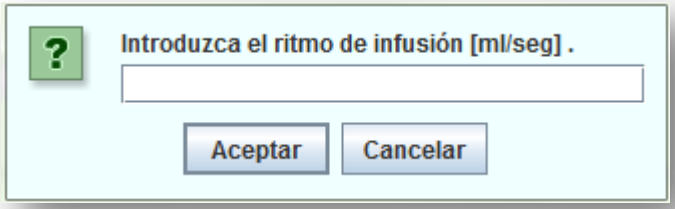



Figura 3.3.26. Función 4.5 Calcular R.out

Para calcular R.out es necesario introducir el ritmo de infusión, por tanto una vez marcadas la basal y onda arriba, aparecerá por pantalla un mensaje pidiendo esta información.

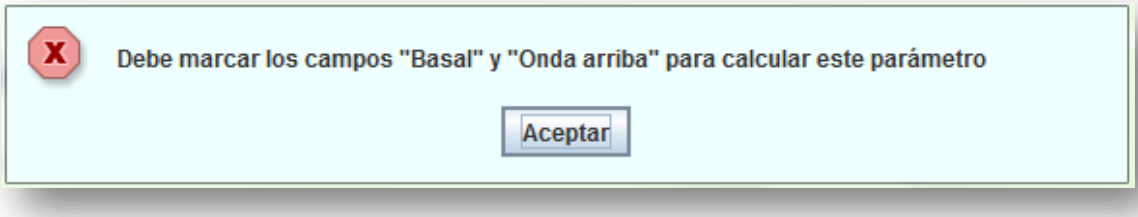



Algoritmo cálculo R.out :

(Onda arriba – Basal) / Ritmo de infusión

*Mensajes de error posibles:*

Si alguno de los valores necesarios (basal u onda arriba) no se han marcado anteriormente, aparecerá por pantalla un mensaje informando de esto.




*Componentes gráficos utilizados:*

- CO.4.5.01 Botón: “CALCULAR R.out”
  Componente Java: JButton
- CO.4.5.02 Campo de texto (calcular R.out)
  Componente Java: JTextField

**Función 4.6 Ver fichero de eventos**

Normalmente basta con ver gráficamente donde ha sucedido el evento, pues son tramos de curva que se descartan en el reconocimiento patrones.

No obstante, por si fuera de interés, esta función permite ver el contenido del fichero de eventos en un área de texto habilitada para ello. Así podremos ver los detalles de cada evento si es necesario.

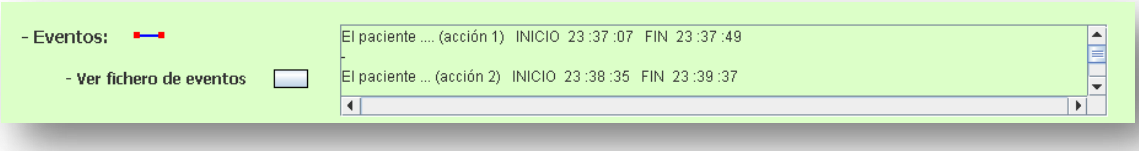



Figura 3.3.27. Función 4.6 Ver fichero de eventos

*Mensajes de error posibles:*

No lanza mensajes de error.

*Componentes gráficos utilizados:*

- CO.4.6.01 Etiqueta: " Eventos: Ver fichero de eventos"
  Componente Java: JLabel
- CO.4.6.02 Botón acceso
  Componente Java: JButton
- CO.4.6.03 Área para ver el contenido del fichero
  Componente Java: JTextArea
- CO.4.6.04 Contenedor con barras de desplazamiento para el JTextArea anterior
  Componente Java: JScrollPane

# 4. PRUEBAS

# Introducción

En el apartado anterior  hemos explicado el diseño y desarrollo de la aplicación. A continuación, en este apartado, comprobaremos su funcionamiento. En primer lugar comprobamos que toda la parte de diseño funcional de la aplicación responde según lo esperado, es decir todas funciones cumplen su cometido.

- 4.1 PRIMERA FASE: PRUEBAS FUNCIONALES DE LA APLICACIÓN

La única función que queda por comprobar es la representación de la PIC en tiempo real y es lo que comprobaremos a continuación; la recepción de datos por puerto serie.

Lo hemos hecho en dos fases, las cuales están recogidas en los siguientes documentos:

- 4.2 SEGUNDA FASE: PRUEBAS CON HYPERTERMINAL
- 4.3 TERCERA FASE: PRUEBAS EN EL HOSPITAL

Comprobado todo lo anterior, tenemos la aplicación funcionando y adaptada al equipo de monitorización de la PIC.

PRUEBAS EN PACIENTES

Por tratarse de software médico quedaría una última fase de pruebas, que no es objeto de este proyecto.

Los equipos médicos que llevan incorporado software médico, deben diseñarse de forma que se garantice la repetitividad, fiabilidad y eficacia de estos sistemas. Si ocurrieran fallos del sistema, deben preverse los medios para poder eliminar o reducir los riesgos consiguientes. Para ello, debe llevarse a cabo un análisis de riesgos de la aplicación, mediante una serie de pruebas debidamente documentadas y diseñadas para este fin.

Como hemos dicho, en este proyecto no vamos a llegar a este tipo de pruebas finales sino que nos limitamos a comprobar que la aplicación funciona correctamente y se adapta al equipo de monitorización.

## 4.1 PRIMERA FASE: PRUEBAS FUNCIONALES DE LA APLICACIÓN

En este apartado resumimos los resultados de las pruebas de cada una de las funciones que realiza la aplicación, a excepción de la función 3.1, monitorización de la PIC, que es objeto de los siguientes apartados.

| FUNCIÓN | PRUEBA | FUNCIONAMIENTO |
|---|---|---|
| **Función 1.1 Acceso protegido con contraseña.** | ok | Introducido el usuario y contraseña correctos da paso a la siguiente pantalla. Si no, aparece por pantalla un mensaje de error. |
| **Función 2.1 Buscador de pacientes.** | ok | Según vamos tecleando caracteres en el buscador aparece debajo un desplegable con una selección de nombres de los pacientes registrados que más se ajustan a lo que buscamos. |
| **Función 2.2 Inicio de monitorización.** | ok | Da paso a la pantalla 3, dedicada a la monitorización de la PIC actual. Previamente comprueba si el paciente es de nuevo ingreso o no. En el caso de que sea nuevo ingreso crea un directorio nuevo para éste. |
| **Función 2.3 Ver registros antiguos.** | ok | Muestra en un desplegable los registros antiguos del paciente seleccionado.<br>Una vez seleccionado el registro, éste se abre en la pantalla 4, pantalla de estudio de la curva. |
| **Función 2.4 Calibrar pantalla.** | ok | Calibra la pantalla según la medida de un centímetro seleccionada. |
| **Función 3.1 Monitorización de la PIC.** | ok | Pruebas detalladas en el apartado 4.2 y 4.3 |
| **Función 3.2 Registros normales separados de registros de infusión** | ok | Nombra los ficheros acorde al tipo de monitorización que se esté realizando. También se mostrará en el encabezado de la pantalla el nombre del paciente junto al tipo de monitorización. |
| **Función 3.3 Marcar manualmente eventos.** | ok | Recoge el nombre del evento, hora de inicio y fin, y lo escribe en el fichero de eventos correspondiente. Los tramos de curva sucedidos durante un evento se pintan en azul. |
| **Función 3.4 Finalizar monitorización** | ok | Al pulsar este botón se finaliza el registro actual, es decir, se cierra la conexión con el puerto serie y se cierran los ficheros en los que se estaban escribiendo los datos recibidos. |
| **Función 3.5 Acceder a la pantalla de estudio de la curva para el registro** | ok | Abre el registro actual en la ventana de estudio de la curva (pantalla 4). |

| | | |
|---|---|---|
| **actual** | | |
| **Función 3.6 Acceder a la pantalla de estudio de la curva para un registro antiguo** | ok | Una vez elegido el registro antiguo de un desplegable, se abrirá en la ventana de estudio de la curva (pantalla 4). |
| **Función 4.1 Mostrar el registro seleccionado.** | ok | Representar la curva de la PIC en la pantalla 4, pantalla de estudio de la curva. |
| **Función 4.2 Aumentar/Disminuir la escala de la PIC** | ok | Aumenta y disminuye la escala de PIC de la curva representada. |
| **Función 4.3 Aumentar/Disminuir la escala temporal** | ok | Aumenta y disminuye la escala temporal de la gráfica de la PIC representada. |
| **Función 4.4 Marcar basal/ onda arriba** | ok | Marcar el valor de la basal u onda arriba directamente sobre la pantalla de la gráfica. El valor aparecerá en el campo de texto situado junto a estos. |
| **Función 4.5 Calcular R.out** | ok | Una vez marcados los valores de basal y onda arriba, pide por pantalla el ritmo de infusión, y calcula el valor de R.out.<br>El valor aparecerá en el campo de texto situado junto al botón. |
| **Función 4.6 Ver fichero de eventos** | ok | Ver el contenido del fichero de eventos en un área de texto habilitada para ello. |

## 4.2 SEGUNDA FASE: PRUEBAS CON HYPERTERMINAL

Ya que el equipo de monitorización de la PIC no se puede sacar del hospital, vamos a comprobar que la recepción de datos por puerto serie es correcta, para ello, vamos a utilizar el hyperterminal de comunicaciones de Windows, con el que simularemos una transmisión de datos.

### QUÉ ES EL HYPERTERMINAL DE WINDOWS

HyperTerminal es un software de comunicaciones utilizado para conectarse a otros equipos a través de módems, conexiones serie RS-232 , o telnet. Con el fin de utilizar hyperterminal, el usuario tendrá que conocer detalles sobre el equipo al que deseen conectarse, tales como el número para marcar, dirección IP o protocolo de comunicaciones (en caso de conectar por puerto serie).

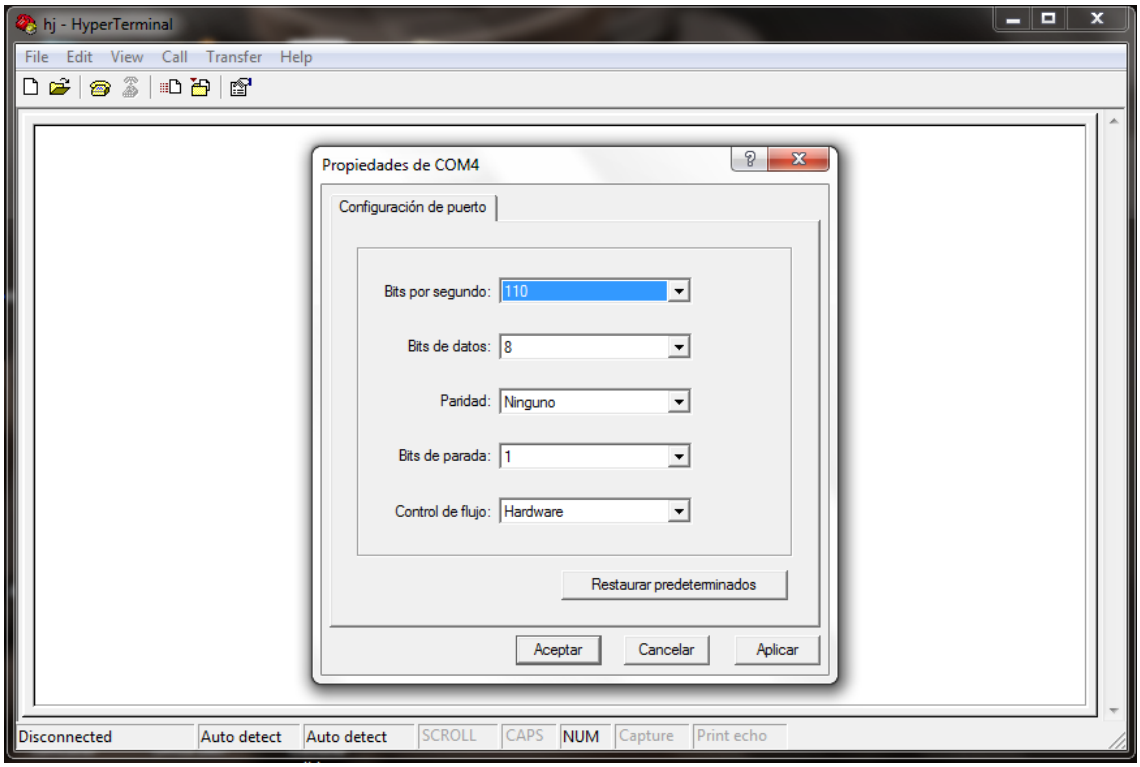
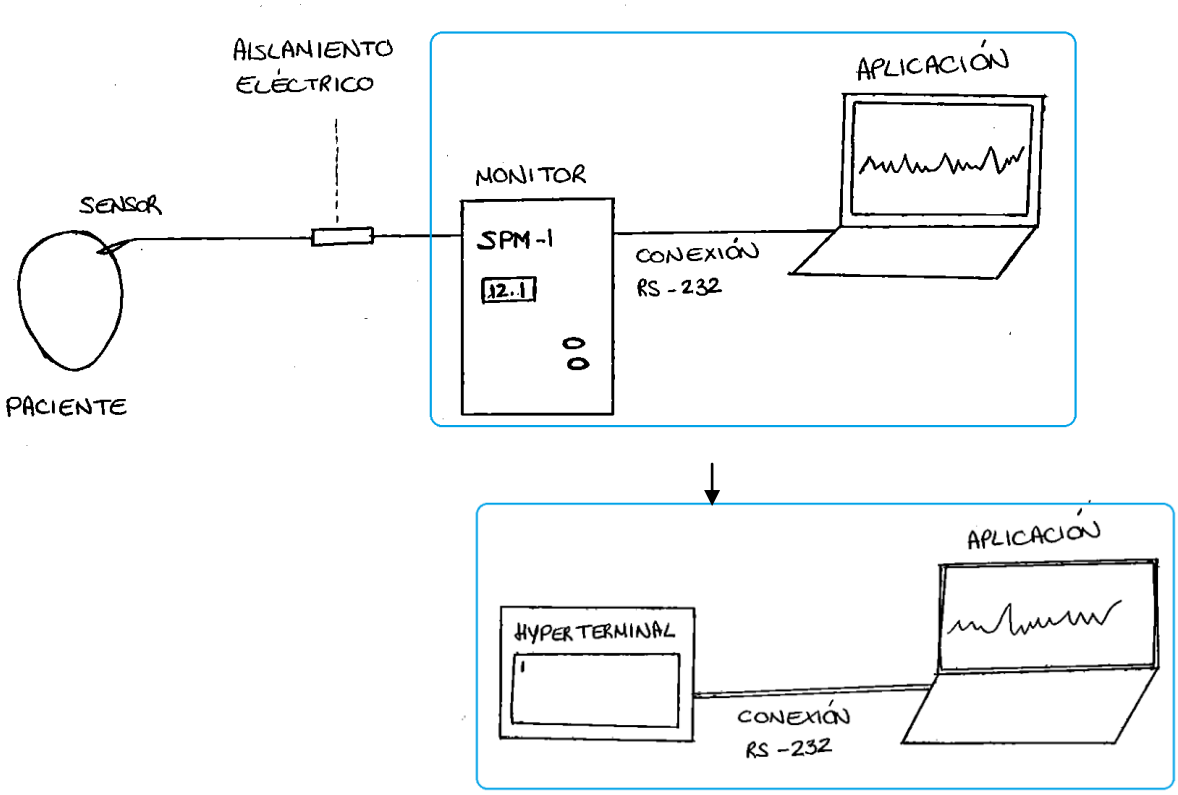



Figura 4.2.1. Hyperterminal de Windows, selección de parámetros para comunicación por puerto serie.

La idea es que es un accesorio estándar de Windows, pero en las versiones más recientes es probable que no lo encontremos, en ese caso es posible descargarlo e instalarlo. Existe una versión mejorada del estándar, HyperTerminal PE.

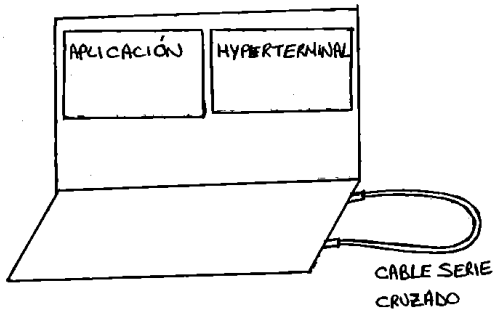
## CONEXIÓN CON HYPERTERMINAL

En nuestro caso vamos a utilizar hyperterminal para simular la conexión con el monitor del equipo del hospital. Esta conexión, es una conexión serie RS-232.




De esta manera la aplicación establecerá comunicación con el hyperterminal (en vez de con el monitor).

No se necesita un PC  adicional para abrir el hyperterminal, sino que se puede hacer desde el mismo ordenador en el que tenemos ejecutando la aplicación.



***PASOS PARA HACER LA CONEXIÓN SERIE ENTRE EL HYPERTERMINAL Y LA APLICACIÓN***

1. **El ordenador tiene varios puertos de entrada/salida de datos (USB, puerto serie…). Conectamos cada extremo del cable RS-232 a un puerto serie.**

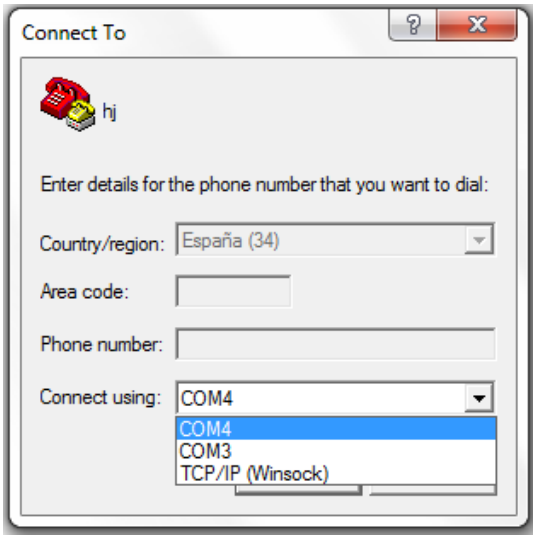
**RS232**: Recommended Standard 232, es una interfaz que designa una norma para el intercambio serie de datos binarios entre un equipo terminal de datos y un equipo de comunicación de datos.
La interfaz RS-232 está diseñada para imprimir documentos para distancias cortas, de hasta 15 metros según la norma, y para velocidades de comunicación bajas, de no más de 20 Kb /segundo.

- Uso de adaptadores.
  En la mayoría de los casos, los ordenadores portátiles no disponen de dos puertos serie RS-232 para este tipo de conexión.
  Esto lo solucionamos utilizando adaptadores, de esta manera conectaremos el cable serie a un puerto USB (más común en portátiles) con un adaptador USB-serie.

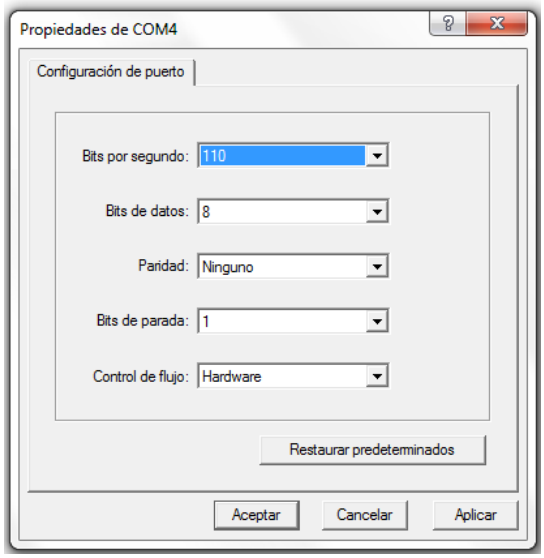
2. **Abrimos y asignamos un puerto al Hyperterminal.**

Al iniciar una comunicación con el Hyperterminal debemos elegir el tipo de conexión que queremos hacer, elegimos transmitir por un puerto serie (COM).

En este caso tenemos dos puertos preparados para transmitir/recibir (COM3, COM4), son los puertos en los que está conectado el cable, elegimos uno de ellos.



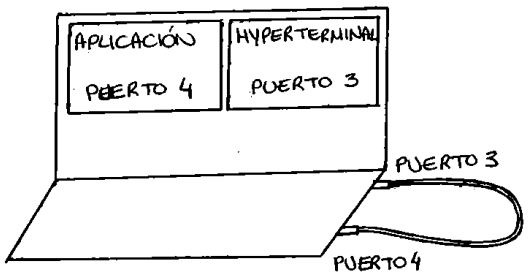

Hecho esto seleccionaremos  los parámetros de la comunicación, y ya tenemos el Hyperterminal conectado y preparado para transmitir.




### 3. Abrimos y asignamos un  puerto a la aplicación

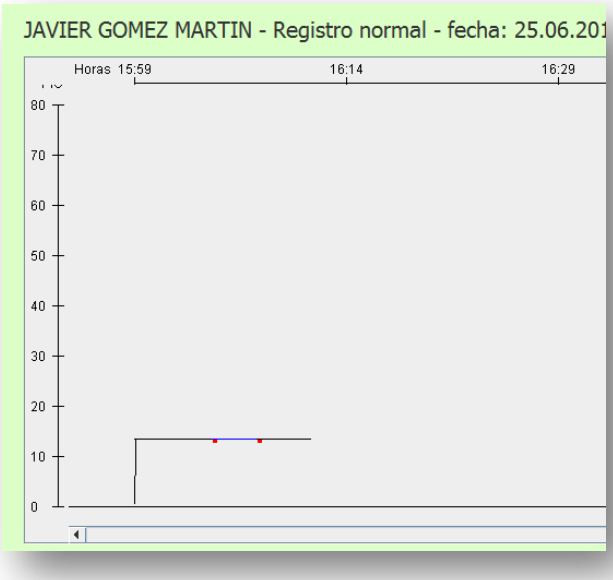
En este caso la aplicación buscará automáticamente los puertos libres para transmitir/recibir, y encontrará uno, donde está conectado el otro extremo del cable (en este caso COM3).

Ajustar los parámetros de la comunicación, para que coincidan en este caso con los del Hyperterminal, y ya tenemos la aplicación preparada para recibir.




## EJECUCIÓN Y RESULTADOS

En el hyperterminal podemos introducir valores manualmente, simulando los valores de la PIC, y estos se enviarán a la aplicación que deberá ser capaz de recibirlos, escribirlos en el fichero correspondiente y representarlos por pantalla atendiendo a las circunstancias.




- ✓ Recibir valores: OK
- ✓ Escribir valores en el fichero correspondiente: OK
- ✓ Distinguir si el valor se ha recibido durante un evento: OK
- ✓ Representarlos por pantalla: OK

## *LIMITACIONES DE LAS PRUEBAS CON HYPERTERMINAL*

El Hyperterminal nos permite comprobar el funcionamiento de la aplicación, pero tiene limitaciones ya que no puede reproducir con exactitud el comportamiento del monitor, ni, en general, del equipo de monitorización del hospital.

Alguna de estas limitaciones son: la velocidad de transmisión de los datos, los parámetros con los que se establece la comunicación (ya que no los sabemos aún), o el formato de los valores transmitidos de la PIC.

## 4.3 TERCERA FASE: PRUEBAS EN EL HOSPITAL

Como hemos mencionado anteriormente el equipo de monitorización de la PIC no se puede sacar del hospital, y por tanto esta parte se ha tenido que trabajar allí.

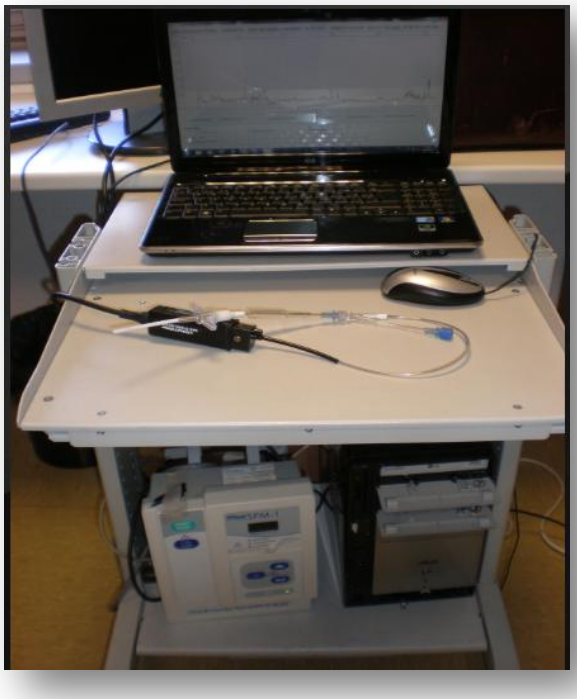


Figura 4.3.1. Equipo completo de monitorización de PIC

Para que la aplicación pueda interpretar los datos que envía el monitor es necesario conocer el protocolo con el que transmite éste último. Este protocolo de comunicación, incluye los parámetros de transmisión y el acondicionamiento de datos en tramas.

**PARÁMETROS PARA ESTABLECER LA COMUNICACIÓN:**

"The MPM-1 connects from its female DB-9 serial port to a personal computer (DB-9, DB-25, DIN-8) using a standard RS232 serial modem cable. It uses 7 data bits, odd parity, one stop bit, at 19.2 kBPS"

- Bits de datos : 7
- Paridad : impar
- Bits de parada : 1
- Bits por segundo : 19200

**TRANSMISIÓN DE DATOS:**

Cada valor de PIC se manda en un paquete de datos.

| Byte 1 | Bytes siguientes |
|---|---|

Byte 1: identificador del tipo de paquete, ya que además de los valores de PIC el monitor manda otra información.

Bytes siguientes: valor de la PIC medido. Los valores de la PIC se codifican en ASCII.

**PRUEBAS DE RECEPCIÓN DE DATOS.**

Los paquetes mencionados anteriormente se mandan en tramas. Nos interesan los paquetes con identificador de PIC, por tanto la aplicación leerá byte a byte del puerto serie y filtrará los paquetes cuyo primer byte sea dicho identificador. Una vez obtenido el valor se hará el tratamiento necesario para representarlo por pantalla (Pantalla 3.Pantalla de monitorización).

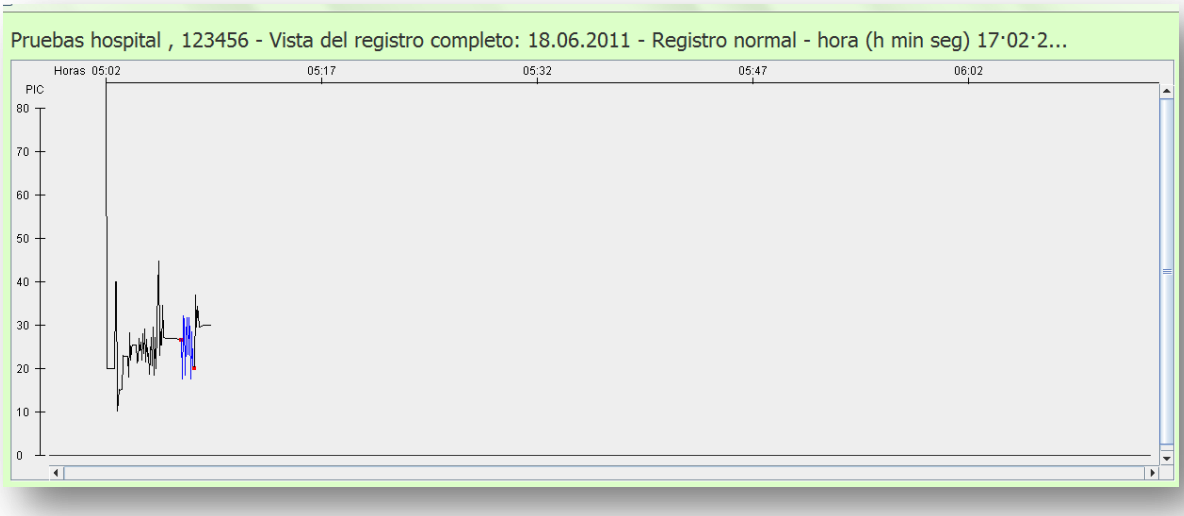
**PRUEBAS DE REFRESCO DE LA PANTALLA**

Debemos comprobar que la aplicación no se demora mucho en hacer el tratamiento de cada valor, y por tanto que la curva que se está representado por pantalla no está desfasada con respecto a los valores actuales.

Las pruebas consisten en variar la presión que mide el sensor y comprobar que la curva refleja estas variaciones instantáneamente.

- Es posible modificar la presión medida gracias al calibrador del sensor. Este calibrador se ajusta manualmente para que el sensor devuelva PIC=0 antes de ser introducido en el ventrículo de un paciente. Una vez introducido el sensor, el calibrador no debe moverse bajo ninguna circunstancia.
  Para las pruebas se ha ido ajustando este calibrador manualmente para que devuelva diversos valores.

Variación de la PIC en la pantalla:



Hechas las pruebas, comprobamos que la pantalla reacciona en el momento en que varía la PIC. En esta captura también podemos ver, en azul, un tramo capturado durante un evento.

**ACONDICIONAMIENTO DE VALORES PARA REPRESENTAR CADA HORA DE MONITORIZACIÓN EN 20 CM.**

El equipo de monitorización transmite un valor de PIC cada segundo, lo que implica 3600 valores en una hora. Teniendo en cuenta la resolución del monitor del PC, tenemos que calcular el número de píxeles existentes en 20 cm de pantalla.

En este caso, en 20 cm de pantalla tenemos 900 píxeles, y por tanto, tenemos que hacer un diezmado de las muestras recogidas, es decir nos quedaremos con una de cada cuatro (3600/900=4).

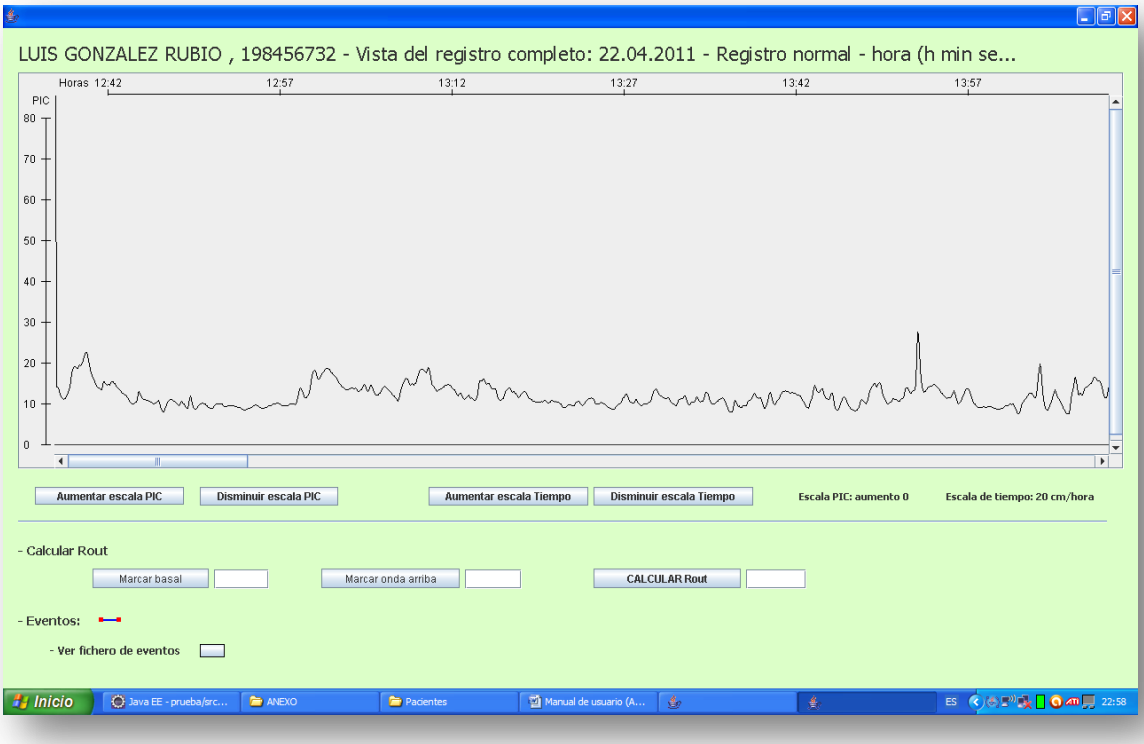
Este diezmado sólo afectará a la representación en tiempo real , ya que en los ficheros del paciente se guardarán todas muestras.

# 5. RESULTADOS

# 5. RESULTADOS

Una vez finalizado el periodo de pruebas, y comprobado que todo funciona correctamente, vamos a verificar que se ha cumplido el objetivo en este documento.

Hemos creado una aplicación nueva para monitorizar la PIC, que cumple con todos los requisitos de usuario y compatible con el equipo de monitorización actual. Esta aplicación permite detectar patrones de hidrocefalia normotensa, gracias al cumplimiento del requisito RU. 06.




Esta aplicación introduce las siguientes mejoras con respecto a la anterior:

- Registros de infusión separados de registros normales (Requisito de usuario 03, RU.03)
- Posibilidad de marcar eventos manualmente (RU.04)
- Posibilidad de marcar basal, onda arriba y calcular R.out sobre la gráfica (RU.05)
- Posibilidad de aumentar/disminuir la gráfica de la PIC (RU .06)
- Posibilidad de calibrar la pantalla según el monitor del PC (RU.07)
- Posibilidad de registrar nuevos pacientes y ver pacientes registrados (RU.08)
- Posibilidad de abrir registros antiguos para ver la evolución del paciente (RU.09)

Ya que todos los requisitos de usuario se han cumplido, la aplicación es compatible con el equipo de monitorización actual, y todo funciona según lo previsto, podemos decir que hemos cumplido con el objetivo de este proyecto.

# 6. CONCLUSIONES Y LÍNEAS FUTURAS

# 6. CONCLUSIONES Y LÍNEAS FUTURAS

Hemos desarrollado una aplicación cumpliendo el objetivo de este proyecto, es decir, todos los requisitos de usuario se han satisfecho, la aplicación es compatible con el equipo de monitorización actual, y todo funciona según lo previsto.

Como hemos mencionado en el apartado "4.Pruebas", aún quedaría una fase más de pruebas para dejar la aplicación definitivamente incorporada en el equipo médico de monitorización de la PIC del hospital Doce de Octubre de Madrid.

**LÍNEAS FUTURAS.**

- Ya que la aplicación está enfocada al uso práctico por parte de diversos usuarios, siempre se pueden proponer nuevas funcionalidades, no incluidas inicialmente, y que aportarán valor añadido al sistema.
- Hacer un estudio de la posibilidad de almacenar los datos de una forma más adecuada (base de datos), a pesar de las limitaciones que nos plantean los requisitos de software.
- Implementar un módulo que permita el análisis automático de ondas, es decir, que detecte patrones en las curvas de PIC.

# 7. PRESUPUESTO

# 7. PRESUPUESTO

En este capítulo se tratará de ofrecer y estimar el coste real del proyecto, atendiendo a una serie de parámetros cómo son el tiempo empleado en el análisis, diseño e implementación, y recursos utilizados.

Una vez finalizado el mismo se dispone de los parámetros necesarios para realizar estimaciones de los costes lo más reales posibles, y siempre adaptadas al desarrollo y construcción de este tipo de sistemas

ESTIMACIÓN DEL COSTE

La fórmula general que permitirá calcular el coste estimado del proyecto es el resultado de la suma del coste humano (personal que interviene en el desarrollo del proyecto) y coste de material o de recursos.

Se realizará una estimación para cada una de las fases y en función del esfuerzo realizado por el profesional en cuestión.

Cada semana se ha estipulado en 40 horas semanales, repartidas en 5 días laborables de 8 horas de duración. Cada una de las fases tendrá un coste distinto, ya que el esfuerzo y la persona que la desarrolla es de categoría distinta a la que realiza otra fase anterior o posterior. La duración de este proyecto se ha estimado en 7 meses, en concreto en 29 semanas.

| FASE | PRECIO (€/hora) | DURACIÓN (Semanas) | COSTE TOTAL (€) |
|---|---|---|---|
| Análisis | 9.40 | 3 | 1128 |
| Diseño | 12.5 | 4 | 2000 |
| Implementación | 9.40 | 12 | 4512 |
| Pruebas | 9.40 | 5 | 1880 |
| Documentación | 9.40 | 5 | 1880 |
| TOTAL | | | 11400 |

Por otro lado tenemos los costes de material y recursos utilizados.

| DESCRIPCIÓN | Nº UNIDADES | PRECIO UNITARIO (€) | COSTE TOTAL (€) |
|---|---|---|---|
| Portátil | 1 | 1100 | 1100 |
| Monitor SPM-1 | 1 | 12000 | 12000 |
| Cable serie cruzado | 1 | 3 | 4 |
| Adaptadores usb-serie | 2 | 4 | 8 |
| Sensor PIC Camino | 1 | 560 | 560 |
| TOTAL | | | 13672 |

Por lo tanto, sumando los costes de personal  y los costes de recursos utilizados el coste total al que asciende el proyecto es de 25072 €.

# 8. BIBLIOGRAFÍA

# 8. BIBLIOGRAFÍA


MANUALES PROGRAMACIÓN

- Aprenda java como si estuviera en primero. Javier Garcia de Jalón, José Ignacio Rodriguez, Iñigo Mingo, Aitor Imaz, Alfonso Brazalez, Alberto Larzabal, Jesus Calleja, Jon García. Año 2000.
- Manual Java Swing. Traducción Juan Antonio Palos http://es.scribd.com/doc/7545519/Manual-JAVA-Swing
- Programación con Swing. Noelia Méndez Fernández. Junio 2005. http://es.scribd.com/doc/7222069/Java-Swing
- Java avanzado. Java Foundation Clases. José m.Ordax. Año 2004. http://www.slideshare.net/POOGAME/swing-2863199

CONSULTAS PUNTUALES

- http://www.chuidiang.com
- http://chuwiki.chuidiang.org
- http://download.oracle.com/javase/1,5.0/docs/api/
- http://www.programacion.com/articulo/swing_y_jfc_java_foundation_classes_94
- http://javaejemplos.com/tutoriales/instalacion-de-jigloo/
- http://java-spain.com/tutorial-jigloo-como-crear-una-interfaz-con-java-y-swing
- http://java-spain.com/tutorial-jigloo-segunda-parte-como-anadir-eventos-y-acciones-swing
- http://java.giovynet.com/Giovynet/
- http://isa.umh.es/asignaturas/sii/p5_java.pdf
- http://download.oracle.com/javase/tutorial/uiswing/layout/group.html
- http://www.gratisweb.com/bshjai/parte6/cap6-2.html
- http://www.magusoft.net/neotrials/pages/java/jpasswordfield.html
- http://www.mailxmail.com/curso-programacion-juegos-moviles-j2me/interfaz-grafica-bajo-nivel
- http://www.java2s.com/Code/Java/Swing-JFC/JScrollPanewithrowandcolumnheaders.htm
- http://www.arrakis.es/~abelp/ApuntesJava/Threads.htm
- http://www.javahispano.org/contenidos/es/graficos_en_java_parte_1/
- http://www.slideshare.net/czelada/hilos-en-java
- http://www.webtaller.com/construccion/lenguajes/java/lecciones/clase-arraylist-java.php
- http://monillo007.blogspot.com/2009/04/leer-la-entrada-de-un-puerto-serial.html

- http://carloszuluaga.wikidot.com/articulos:manejo-de-fechas-en-java-ii
- http://tabasco.torreingenieria.unam.mx/GCH/Curso%20de%20Java%20CD/Documentos/froufe/parte15/cap15-3.html
- http://dis.um.es/~bmoros/Tutorial/parte14/cap14-17.html


ARTÍCULOS MÉDICOS


- http://neuroc99.sld.cu/text/hipertensionendoc.htm
- http://www.neurocirugia.com/diagnostico/hidrocnormo/hidrocefnormo.htm
- http://www.nlm.nih.gov/medlineplus/spanish/ency/article/000752.htm
- http://www.infodoctor.org/www/hidrocefaliapresionnormal.htm
- http://www.nlm.nih.gov/medlineplus/spanish/ency/article/003411.htm
- http://www.cirujanosdechile.cl/Revista/PDF%20Cirujanos%202004_06/Rev.Cir.6.04.(03).AV.pdf

# 9. ANEXO

# MANUAL DE USUARIO

# MANUAL DE USUARIO

La aplicación “Med PIC” está diseñada para que su uso sea lo más intuitivo posible, en caso de duda consulte el apartado correspondiente en este manual.

### Pantalla Bienvenida

*Introducir contraseña*

### Pantalla Registro

*Buscar un paciente*

*Iniciar un registro*

*Ver un registro antiguo*

*Calibrar pantalla del monitor*

### Pantalla monitorización dinámica

*Iniciar/ finalizar una monitorización*

*Marcar eventos manualmente*

*Acceder a la pantalla de estudio de la curva desde aquí*

*Abrir un registro antiguo desde aquí*

### Pantalla de estudio de la curva

*Desde dónde se puede acceder a esta pantalla*

*Niveles de aumento*

*Marcar basal/onda arriba y calcular R.out*

*Ver fichero de eventos*

### PANTALLA BIENVENIDA

El propósito de esta pantalla de inicio es dar la bienvenida al usuario y proteger el sistema con una contraseña.  Una vez introducida la contraseña correctamente da paso a la siguiente pantalla.

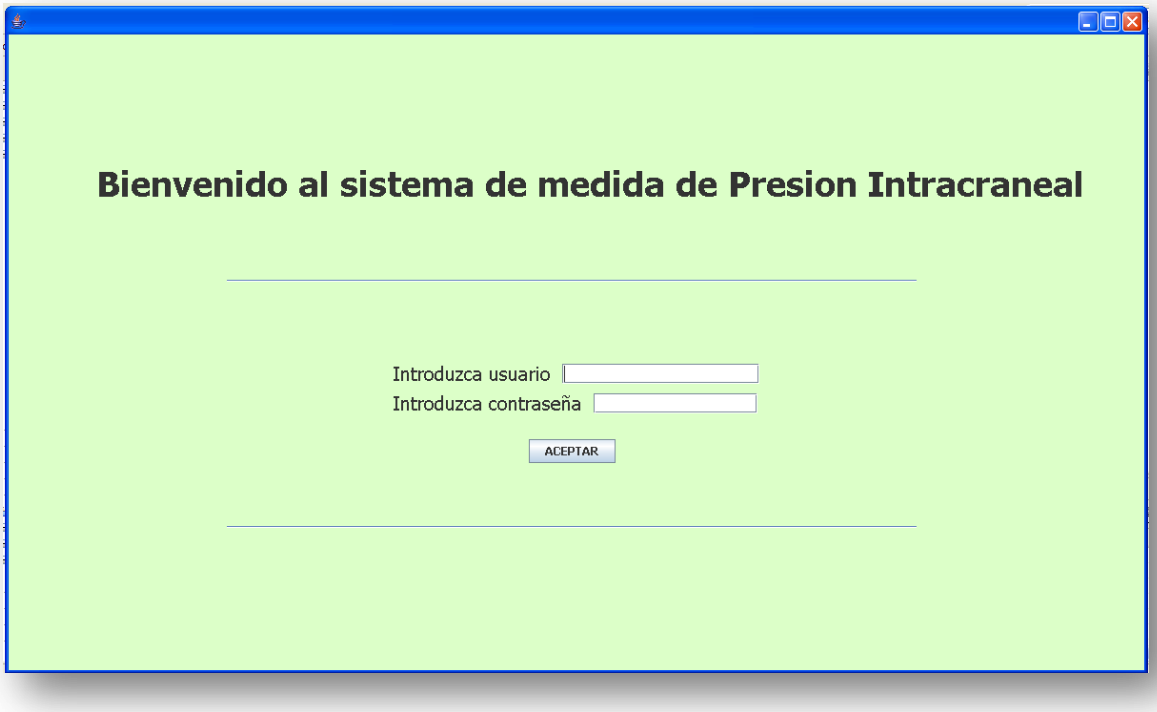
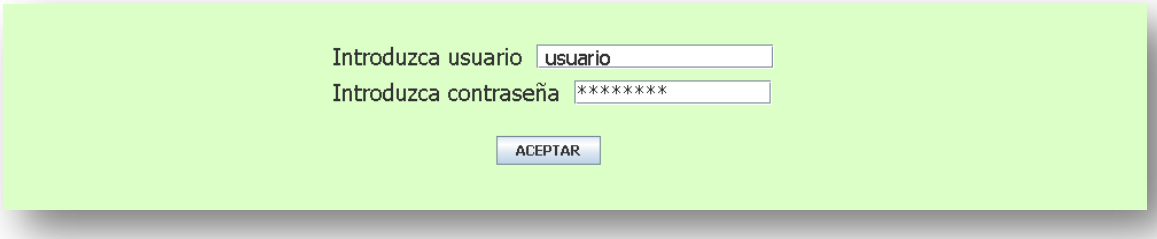



Figura 1. Pantalla de inicio

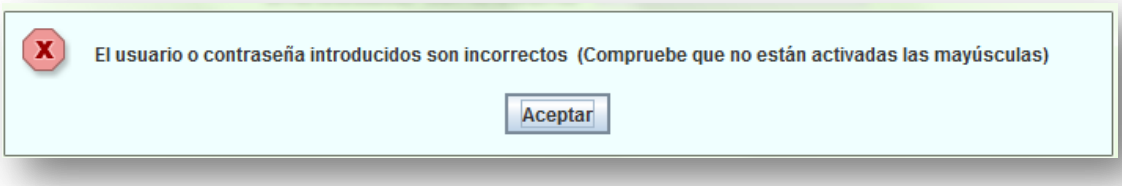
**INTRODUCIR CONTRASEÑA**




Para comenzar a utilizar la aplicación es preciso introducir un nombre de usuario y una contraseña, que en este caso serán el mismo para todo el personal.

- Usuario: usuario
- Contraseña: contraseña

Hecho esto, presionando el botón “ACEPTAR”, dará paso a la siguiente pantalla.
En caso de introducir un usuario o contraseña incorrectos aparecerá el siguiente mensaje:


El usuario o contraseña introducidos son incorrectos (Compruebe que no están activadas las mayúsculas)
Aceptar

## PANTALLA REGISTRO

Esta pantalla tiene dos propósitos  generales, y por tanto está dividida en dos áreas.

- La primera permite el registro/selección de pacientes para iniciar una nueva monitorización o  ver una antigua.
- La segunda  permite la modificación de ciertos parámetros para la calibración del monitor.

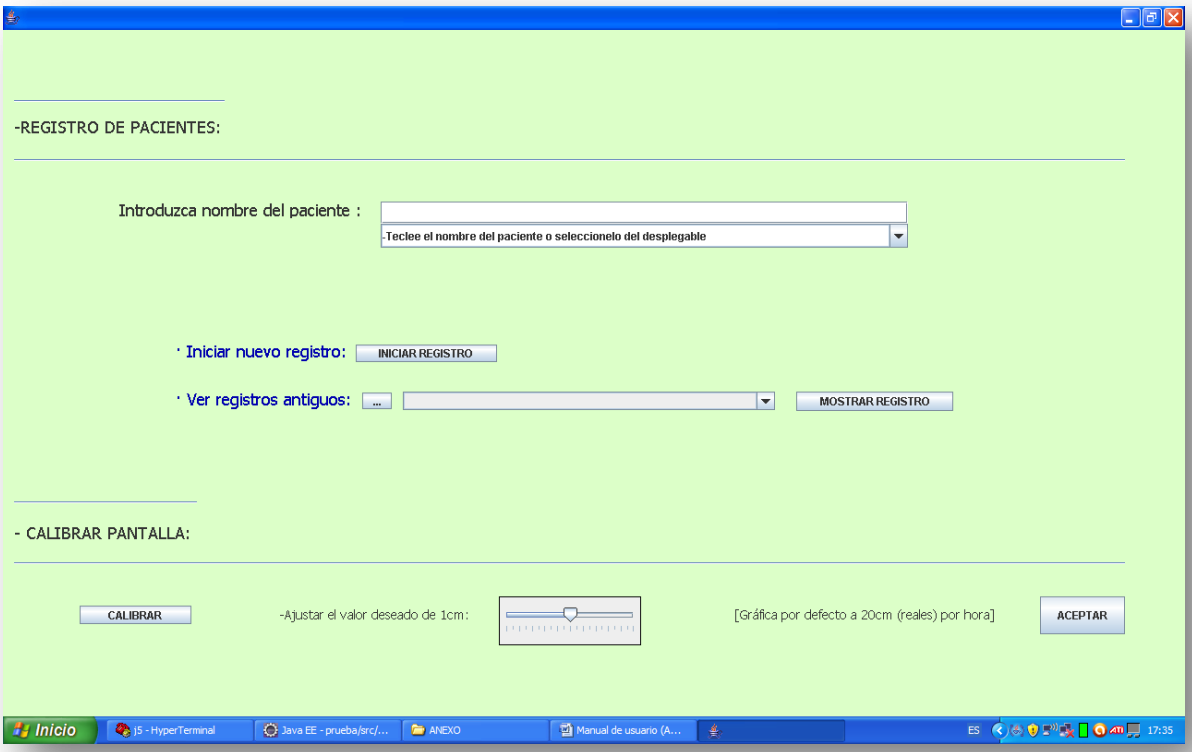
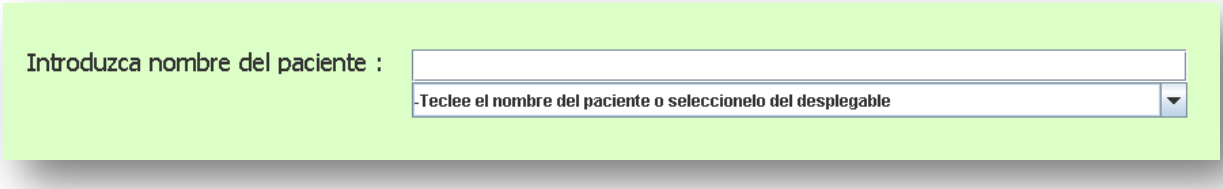
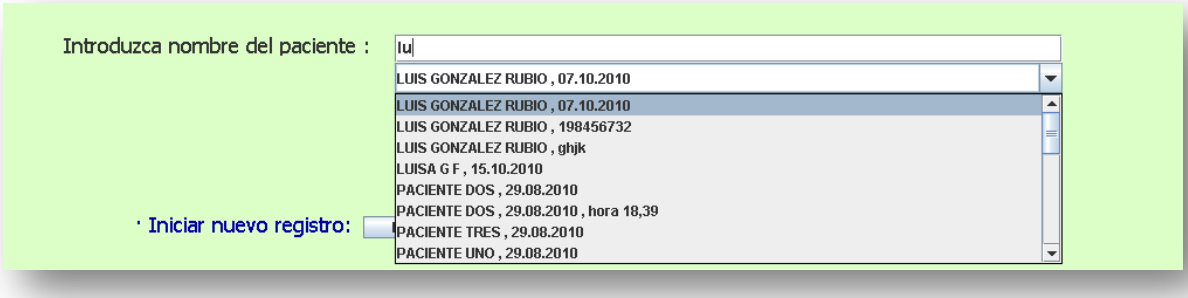



Figura 2 . Pantalla de registro

**BUSCAR UN PACIENTE**




Según vamos tecleando caracteres en este campo aparecerá debajo un desplegable con una selección de nombres de los pacientes registrados que más se ajustan a lo que buscamos.

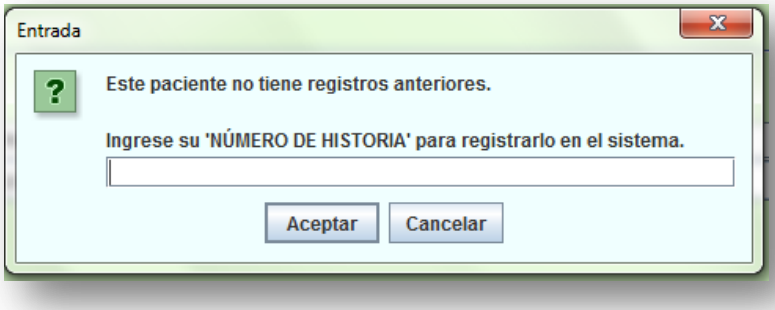


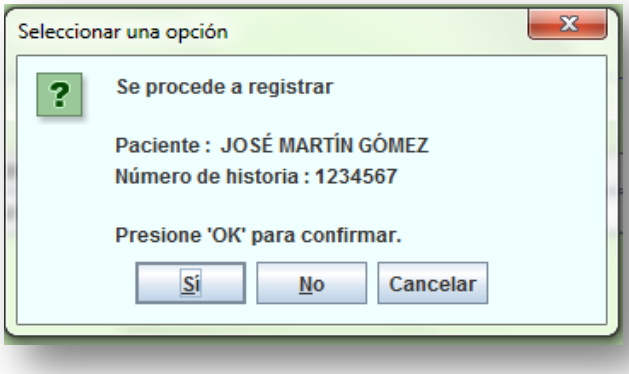

De esta manera, podremos seguir tecleando el nombre del paciente (si es de nuevo ingreso) o seleccionarlo del desplegable.

**INICIAR UN REGISTRO**

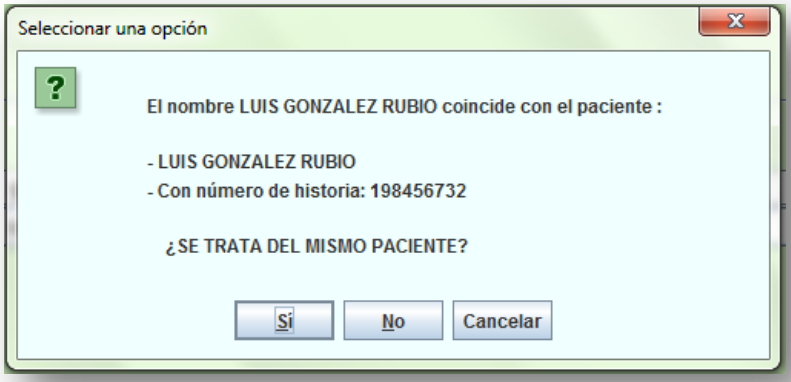

Una vez introducido el nombre del paciente podemos proceder a iniciar un nuevo registro.

Si el paciente es de nuevo ingreso se pedirá por pantalla introducir su número de historia para registrarlo en el sistema.

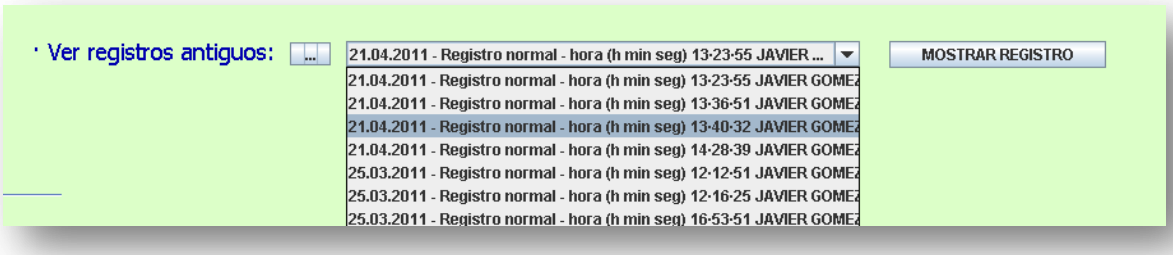


Si  el nombre del paciente coincide con un nombre ya registrado se pedirá confirmación por pantalla (si realmente se trata de un nuevo paciente).



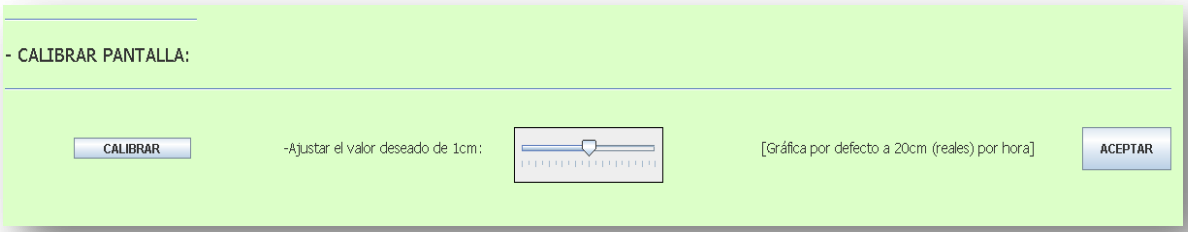

**VER UN REGISTRO ANTIGUO**




Una vez introducido el nombre del paciente podemos proceder a visualizar por pantalla la curva de un registro realizado anteriormente. Los registros se muestran en un desplegable ordenados por fecha, como muestra la figura.

Si el paciente seleccionado no tiene registros antiguos saldrá por pantalla un mensaje como este.

**CALIBRAR PANTALLA DEL MONITOR**




La aplicación da la posibilidad de calibrar la pantalla donde se representa la PIC.

En la pantalla de monitorización la gráfica de la PIC se representa por defecto a 20 cm/hora, pero en ocasiones se dispone de monitores más pequeños y para que este parámetro permanezca inalterable hay que ajustar el valor de un centímetro real.

Para este propósito tenemos este componente. Su uso es sencillo, con una regla de medida sobre el monitor, y en concreto sobre esta barra de desplazamiento, se ajustará la medida de un centímetro real. De esta manera sea cual sea el tamaño del monitor siempre se podrá ajustar la gráfica de la PIC a 20 cm reales/hora

## PANTALLA MONITORIZACIÓN DINÁMICA

Esta es la pantalla donde se representan los datos de la PIC extraídos del sensor en tiempo real. El propósito principal de esta pantalla es, por tanto, mostrar la curva de PIC, además de permitir otras funciones como el registro manual de eventos o el acceso a registros antiguos.

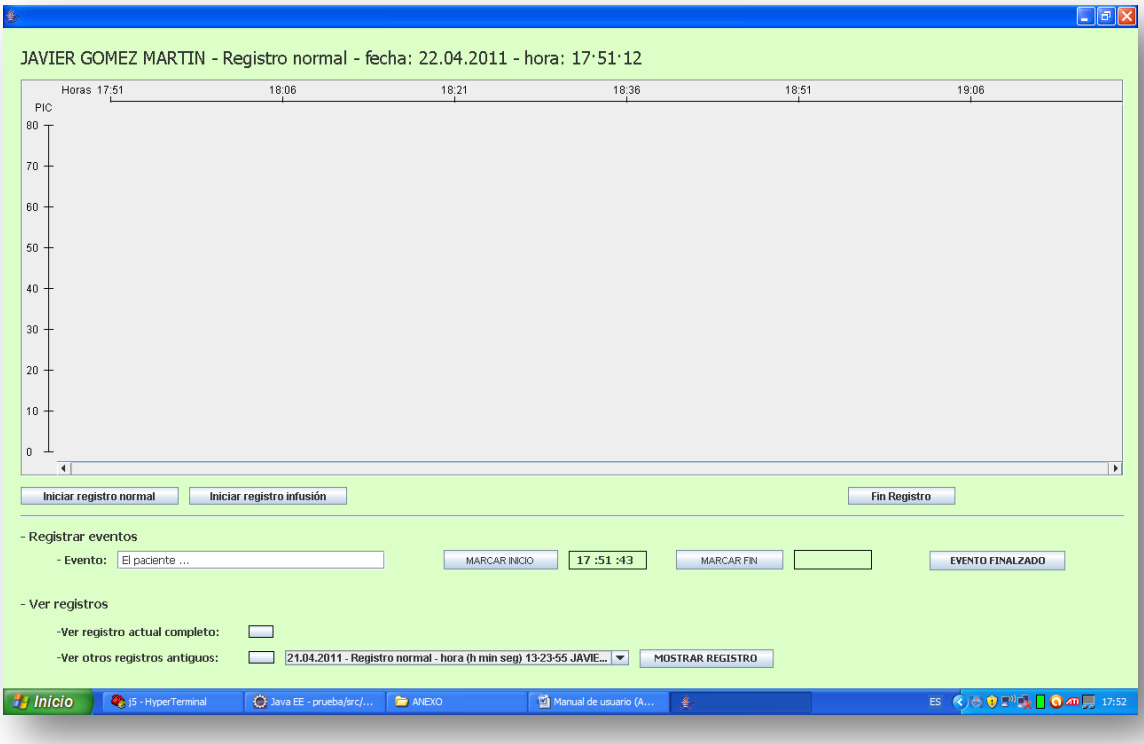
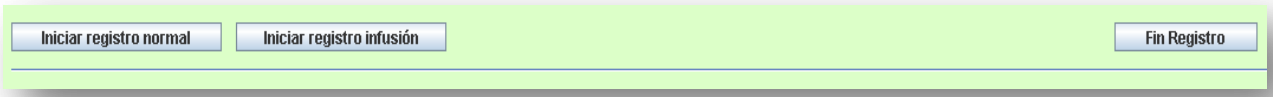



Figura 3.Pantalla de monitorización.

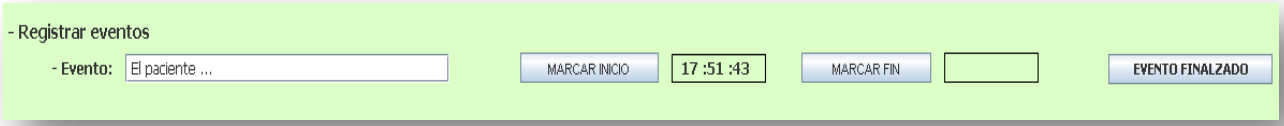
### INICIAR/ FINALIZAR UNA MONITORIZACIÓN




Iniciar una monitorización: es posible iniciar una monitorización normal o un registro de infusión, en el directorio del paciente estos registros aparecerán claramente diferenciados.

Finalizar una monitorización: se dejarán de recoger datos del monitor y por tanto finalizará el registro, que quedará guardado el directorio del paciente para consultas posteriores.

### MARCAR EVENTOS MANUALMENTE



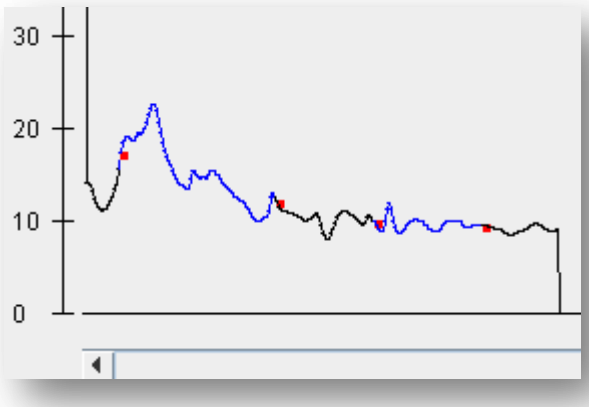

El procedimiento para registrar un evento es el siguiente:

1 Escribir el evento
2 "Marcar inicio": la hora actual aparecerá en el campo de texto.
3 "Marcar fin": la hora actual aparecerá en el campo de texto.

4 "Completado": su función es recoger los datos de los campos anteriores, grabarlos en el fichero de eventos correspondiente y por último borrar los campos de texto (quedando vacios para un nuevo evento).

TRAMOS DE EVENTOS

Los valores recibidos durante un evento aparecerán en azul mientras que los demás se dibujarán en negro.

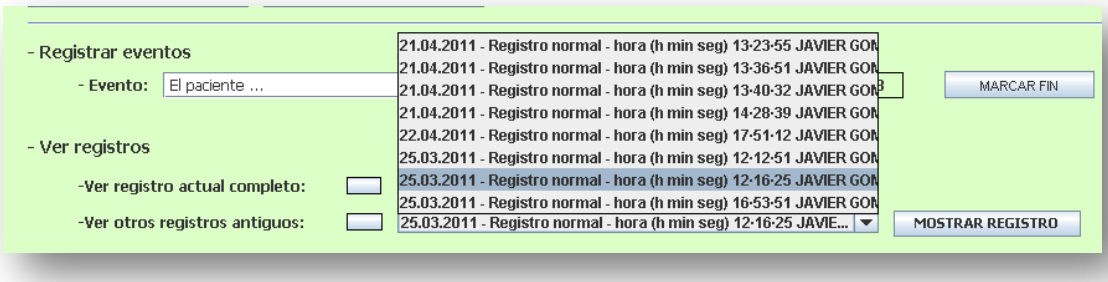



Como podemos ver en la imagen, el comienzo y fin de un evento se marca con un rectángulo rojo, para que se distingan a primera vista estos tramos especiales.

**ACCEDER A LA PANTALLA DE ESTUDIO DE LA CURVA DESDE LA PANTALLA DE MONITORIZACIÓN**

Esta opción abre el registro actual en la ventana de estudio de la curva. Esto es útil para ver en detalle la curva de la PIC que se está monitorizando, y aplicar sobre ella acciones tales como aumentar, disminuir o marcar basal.

**ABRIR UN REGISTRO ANTIGUO DESDE LA PANTALLA DE MONITORIZACIÓN**




Es interesante poder ver otros registros del paciente del que se están cogiendo muestras actualmente, esto permite ver su evolución y detectar patrones de ondas. Los registros se muestran en un desplegable ordenados por fecha, como muestra la figura.

Una vez elegido el registro, se abrirá en la ventana de estudio de la curva.

## PANTALLA DE ESTUDIO DE LA CURVA

El propósito de esta pantalla es mostrar la curva de la PIC (actual o de una monitorización antigua) ofreciendo diversas funciones que permiten su estudio.

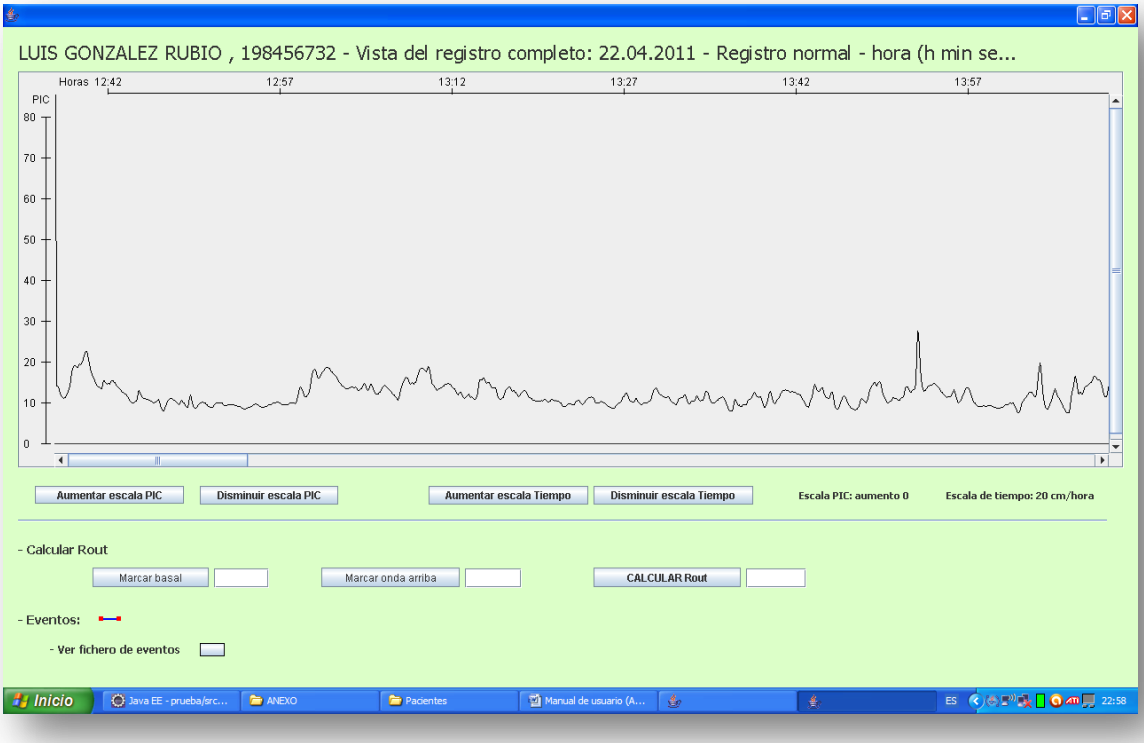



Figura 4. Pantalla de estudio de la curva.

### DESDE DÓNDE SE PUEDE ACCEDER A LA PANTALLA DE ESTUDIO DE LA CURVA

Es posible acceder a esta pantalla desde dos puntos de la aplicación:

- Pantalla de registro > Ver registros antiguos de un paciente seleccionado.
- Pantalla de monitorización > Ver registro antiguos del paciente con registro en curso.

### NIVELES DE AUMENTO

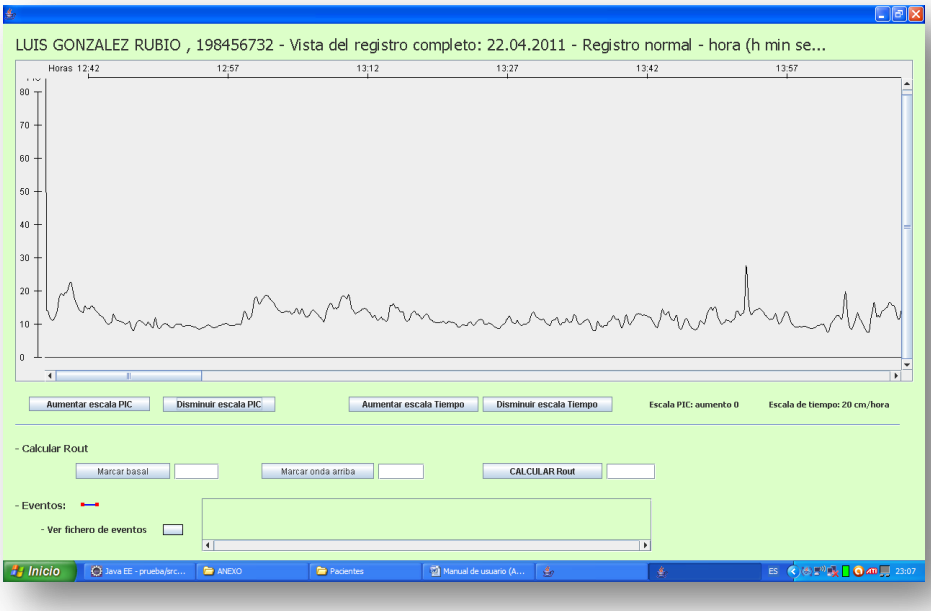
Existe la posibilidad de aumentar/ disminuir la gráfica. Este aumento/disminución se tratará en los ejes, temporal y de amplitud, independientemente. Siendo posible de esta manera un amplio abanico de posibilidades de vista de la gráfica.

- Vistas del eje temporal:
  - 10 cm / hora
  - 20 cm/ hora
  - 40 cm/ hora
  - 60 cm /hora

- Vistas del eje valor de la PIC
  - Aumento 0 → valores de 0 a 80 visibles simultáneamente en la pantalla
  - Aumento 1 → valores de 0 a 80 espaciados con precisión de 5 decimales, visibles simultáneamente 40 valores
  - Aumento 2 → valores de 0 a 80 espaciados con precisión de 1 decimal, visibles simultáneamente 20 valores

*Ejemplo 1*

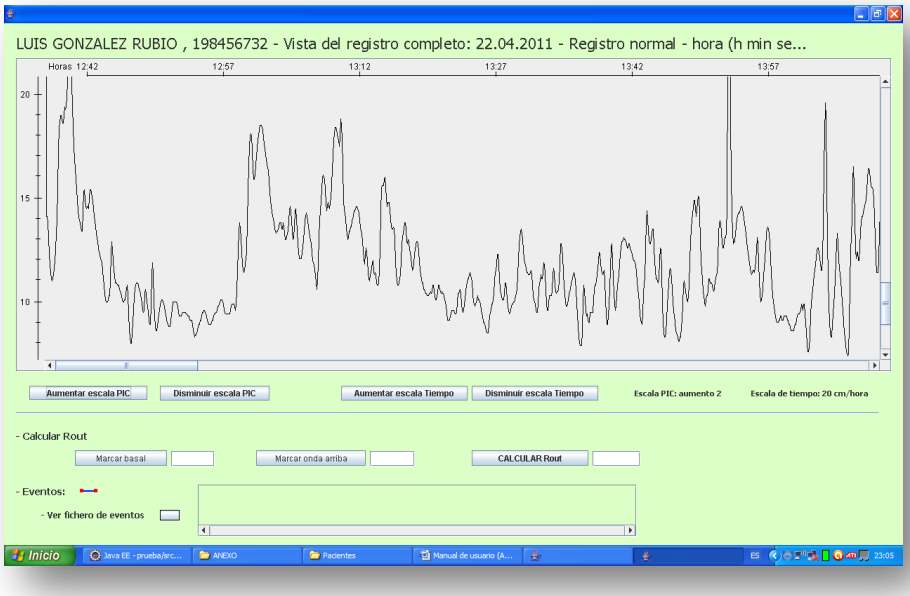
Escala PIC: Aumento 0 Escala temporal: 20cm /hora





*Ejemplo 2*

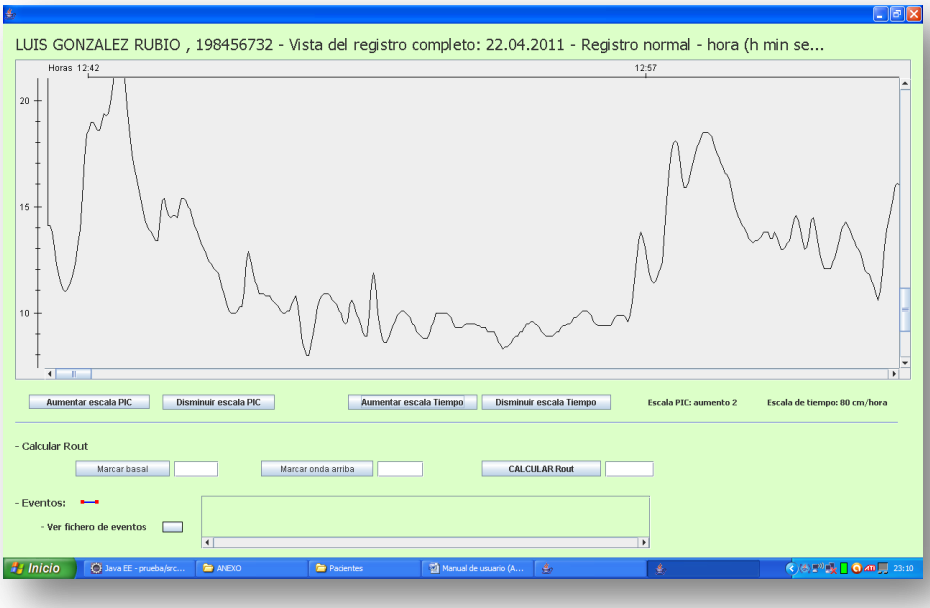
Escala PIC: Aumento 2 Escala temporal: 20cm /hora



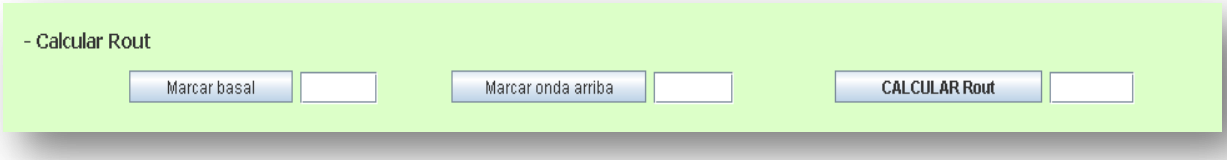


*Ejemplo 3*

Escala PIC: Aumento 2 Escala temporal: 80cm /hora




**MARCAR BASAL/ONDA ARRIBA Y CALCULAR R.OUT**

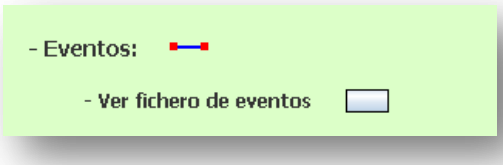



Permite marcar el valor de la basal/onda arriba directamente sobre la pantalla de la gráfica. Una vez pulsado el botón correspondiente y colocado el ratón sobre la gráfica aparecerá una línea horizontal según la posición en que lo vayamos moviendo.
[Dibujo]

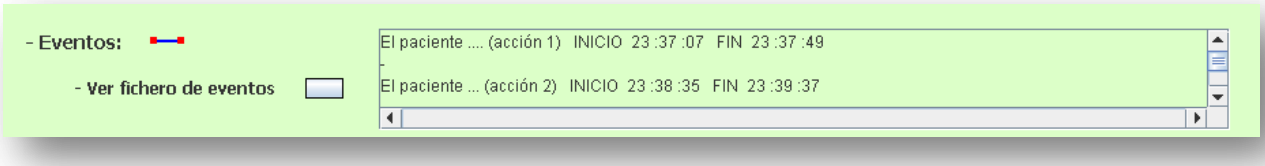
Una vez marcados los valores de basal y onda arriba, podremos calcular la R.out pulsando el botón para dicho efecto. El valor aparecerá en el campo de texto situado junto al botón.

Si alguno de los valores necesarios (basal u onda arriba) no se han marcado anteriormente, aparecerá por pantalla el siguiente mensaje informando de esta situación.

**VER FICHERO DE EVENTOS**




Lee el fichero de eventos correspondiente a la curva que se está visualizando, y muestra su contenido por pantalla en un área específica para este fin.



# INFORMACIÓN DE INTERÉS PARA FUTUROS PROGRAMADORES

# A . MANUAL DE INSTALACIÓN DEL ENTORNO DE PROGRAMACIÓN NECESARIO

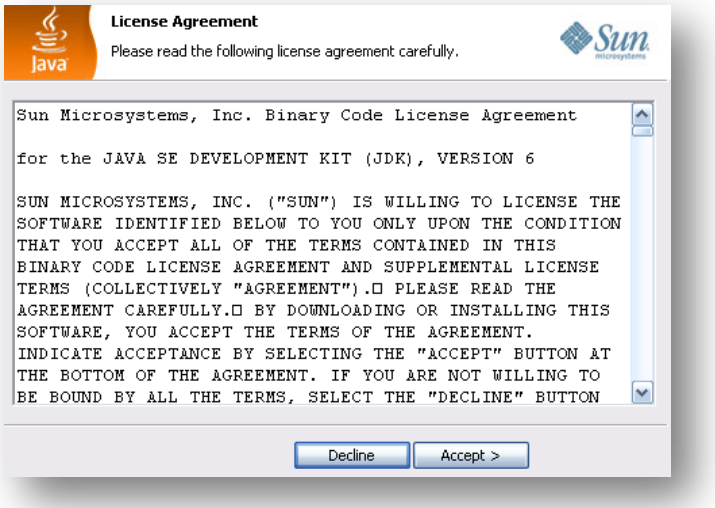
El entorno necesario para escribir, compilar y ejecutar esta aplicación Java es el siguiente:

- J2SE (Java 2 Standard Edition)
- Entorno de Desarrollo Integrado para Java (IDE): Eclipse

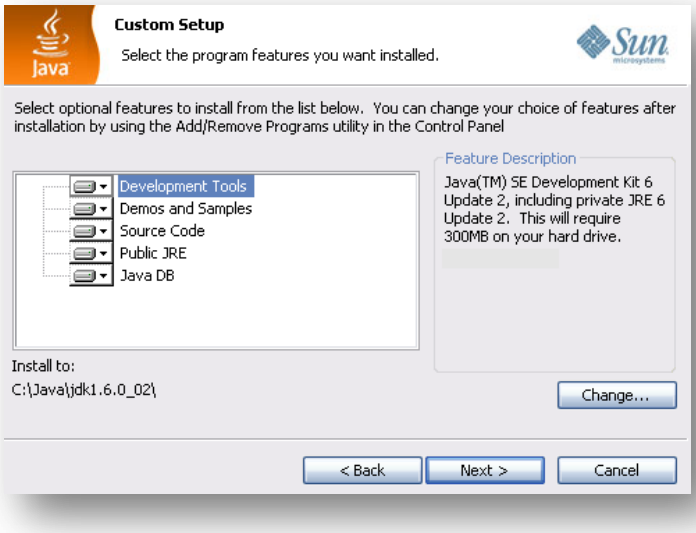
***INSTALACIÓN DE J2SE EN WINDOWS (XP)***

Primero vamos a descargar Java SE Development Kit (JDK), se puede descargar en: http://java.sun.com/javase/downloads/index.jsp

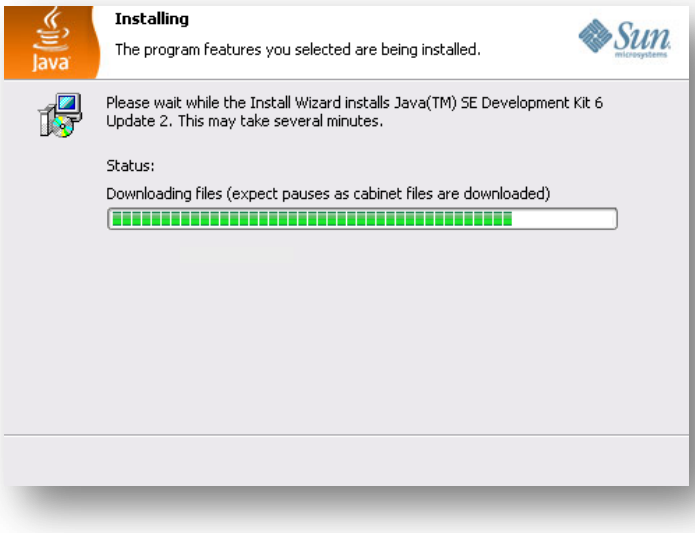
Ejecutamos el archivo que bajamos y aparecerá una pantalla como esta. Damos a aceptar 'Accept'




Nos aparecerá donde queremos que se instale, en este caso esta instalado en C:/Java/ pero se puede dejar que se instale por defecto (C:/Archivos de programa/)



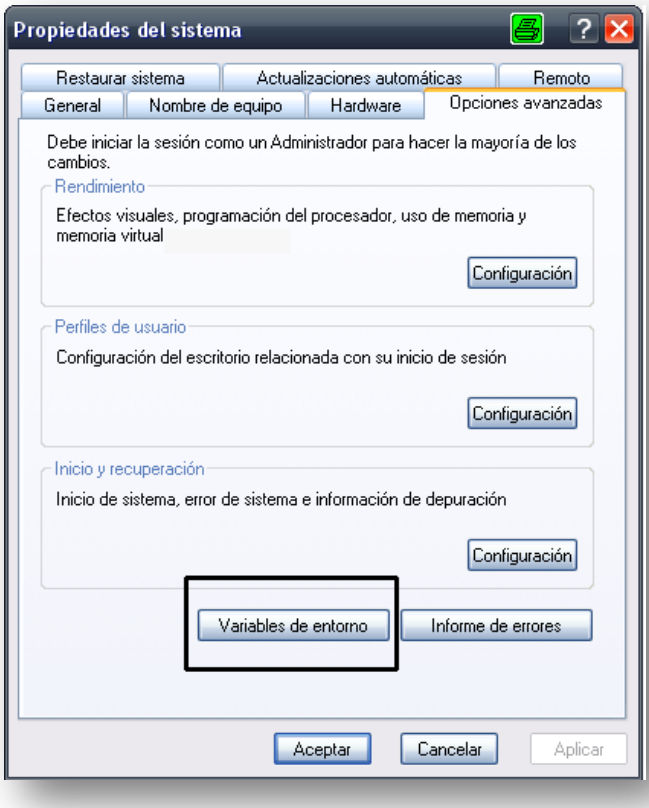

Le damos clic en Next y empezará la instalación de Java Development Tools:




Ahora pedirá donde queremos instalar el Java SE, este es indispensable para Windows XP. En este caso está instalado en C:/Java/, pero se puede dejar la carpeta por defecto, que es C:/Archivos de programas/

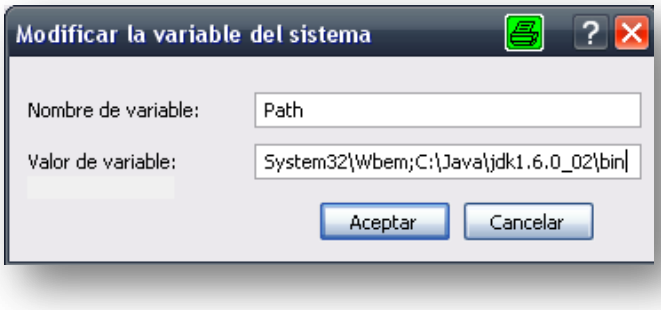
Dando a Next empezará la instalación de Java SE. Terminada la instalación, deberá aparecer una ventana como esta y damos clic en Finish.


Hecho esto, es necesario indicarle al ordenador donde esta el compilador de JAVA, para esto vamos a 'MI PC', hacemos clic con el botón derecho del ratón, vamos a 'Propiedades' y después a la pestaña de 'Opciones Avanzadas'. Pinchamos en 'Variable de entorno'.

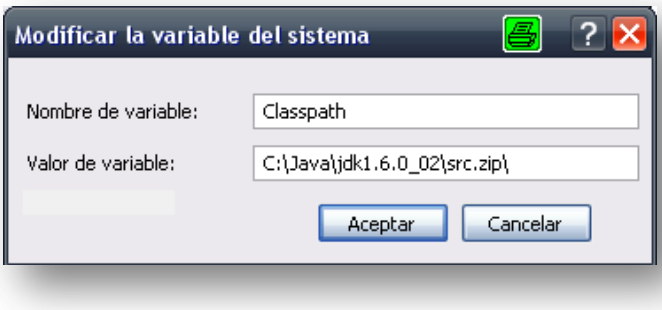



Buscamos la variable llamada "Path" en las Variables del Sistema y pinchamos en modificar. Al final del campo llamado "valor de variable" escribimos la ubicación del compilador de Java, en nuestro

caso escribimos ";C:\Java\jdk1.6.0_02\bin" (el ; que hay antes de C: separa nuestro directorio de los otros valores anteriores)




En la misma ventana de Variable de entorno, creamos una variable llamada 'Classpath', en 'valor de la variable' tendremos que introducir la dirección de la variable. En este caso la dirección sería C:\Java\jdk1.6.0_02\src.zip\ Para terminar hacemos clic en Aceptar y cerramos todas las ventanas.

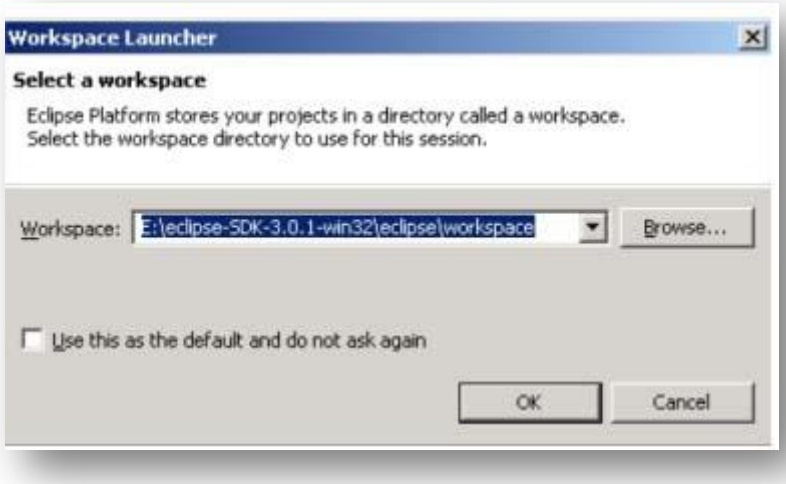



Ahora vamos *al Menu Inicio -> Todos los programas -> Accesorios -> Símbolo del sistema.*
Aparecerá ante nosotros una Consola de DOS, en ella escribimos' *java*' y hacemos un enter. Después tecleamos '*javac'* para ver si el sistema encuentra el intérprete de Java. Si nos sale algún error, hay que volver atrás para revisar la configuración de las variables de entorno del sistema.

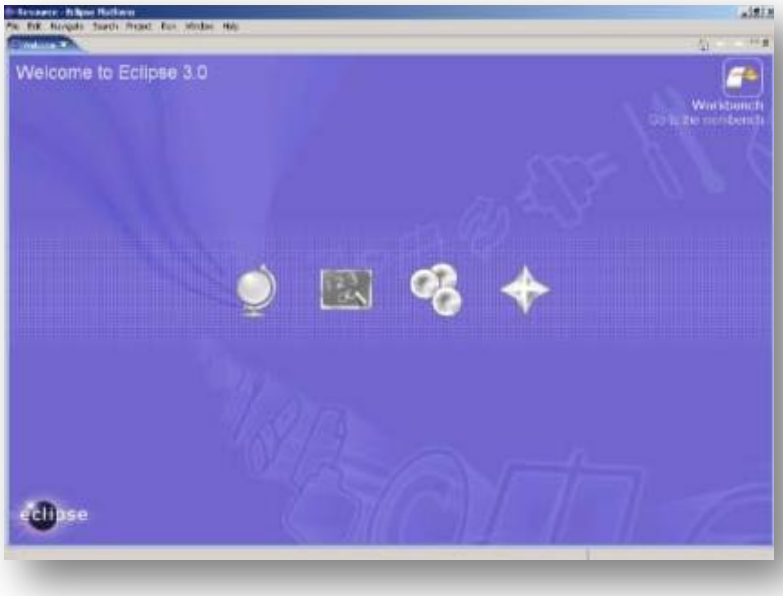
***INSTALACIÓN DE ECLIPSE***

La versión que he utilizado es Eclipse Ganymede.
Podemos descargarlo de www.eclipse.org en forma de archivo ZIP y sólo se debe descomprimir en la carpeta donde queramos tenerlo instalado.

Para ejecutarlo hay que arrancar el fichero Eclipse.exe. Una vez arrancado, nos pedirá que le demos la ruta por defecto donde queramos que Eclipse nos vaya guardando los proyectos que creemos:




Una vez hecho esto lo siguiente que aparecerá es la ventana de inicio de Eclipse:




***INSTALACIÓN PLUG-IN PARA CREAR INTERFACES GRÁFICAS EN ECLIPSE: JIGLOO***

Hay dos formas de instalar plug-ins en Eclipse, dependiendo del plugin:

- Si se da una url (dirección web) para instalar el plugin, desde el mismo Eclipse hay que seguir los siguientes pasos:
  - "Help"
  - "Software updates"
  - "Find and install"
  - "Search for new features to install"
  - "New remote site"

  y a partir de ahí metemos los datos de la url.

- Otros plugins se descargan en un archivo 'zip'. Al descomprimir el archivo 'zip' tiene dentro un directorio plugins y otro features. En el sitio que tenemos instalado Eclipse hay otros dos subdirectorios con los mismos nombres. Basta copiar, con eclipse cerrado, los contenidos de los respectivos directorios, cada uno en su sitio, y arrancar Eclipse.

## B. INTERFAZ GRÁFICA EN JAVA CON SWING

Cómo crear una interfaz gráfica en java utilizando el paquete swing

**LA CONSTRUCCIÓN DE INTERFACES GRÁFICAS DE USUARIO EN JAVA SE DESCOMPONE EN TRES PARTES**

- El contenedor (container), que es la ventana (o parte de la ventana) donde se situarán los componentes.

- Los componentes, que son los menús, botones, áreas de texto etc...

- El modelo de eventos.

  Cada acción producida por el usuario (con el ratón o el teclado), produce un evento que puede ser captado por Java, para permitir ejecutar un código si es necesario. Por defecto, Java no capta ninguno evento sobre ningún objeto. Hay que registrar el evento sobre el objeto para que este sea captado.

**VAMOS A VER CUAL ES EL PROCEDIMIENTO PARA CREAR UNA INTERFAZ GRÁFICA CON UNA VENTANA**

```
public class Ventana extends javax.swing.JFrame{
//declaración de variables
  //Método main
   public static void main (String [] args){
       Ventana  v0 = newVentana();      //Creamos una nueva instancia Ventana
       V0.setLocationRelativeTo (null);
       v0.setVisible (true);                  //La hacemos visible
}
  //Ventana
```

```
  public Ventana {
        super();
        initGUI();                          //Inicializa la ventana
   }
  //Inicializamos la ventana
  private void initGUI(){
  /* 1-Creamos un componente "Contenedor"
     2-Creamos los componentes internos (botones, etc)
     3-Añadimos los componentes internos al "Contenedor*/
  }
  //4-Dotamos de funcionalidad a los componentes
  }
```

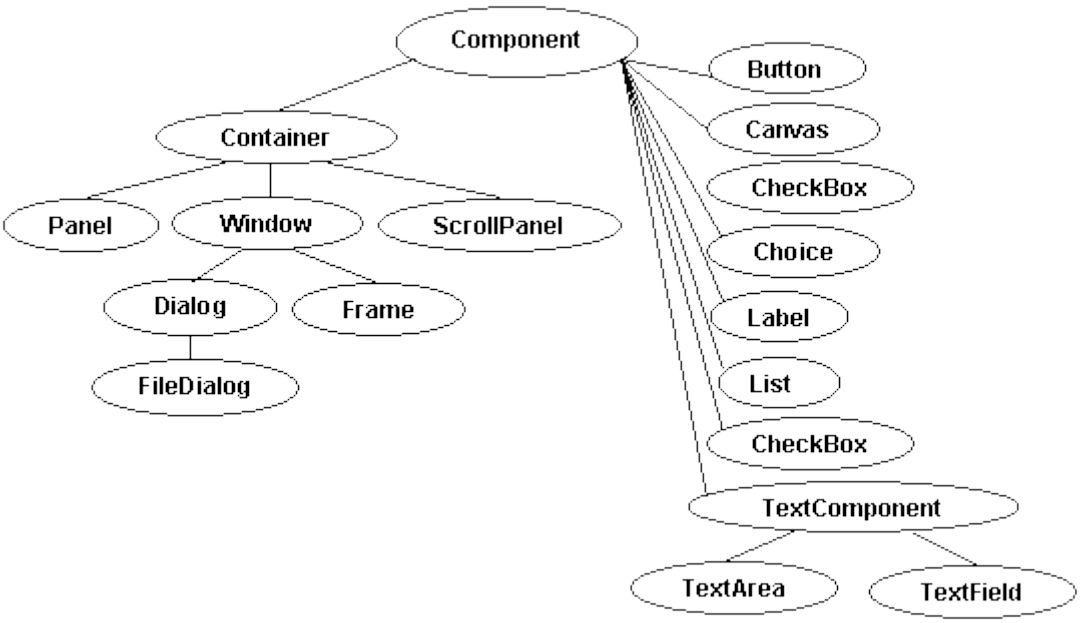
**VAMOS A VER CON MÁS DETALLE ESTOS CUATRO PASOS ANTERIORES**

1.y 2. Creamos un contenedor y sus componentes internos.

Una aplicación Swing se construye mezclando componentes con las siguientes reglas.

- Debe existir, al menos, un contenedor de alto nivel, que provee el soporte que los componentes Swing necesitan para el pintado y el manejo de eventos.
- Otros componentes colgando del contenedor de alto nivel (éstos pueden ser contenedores o componentes simples).

La siguiente figura muestra la jerarquía de clases para las principales clases de AWT:



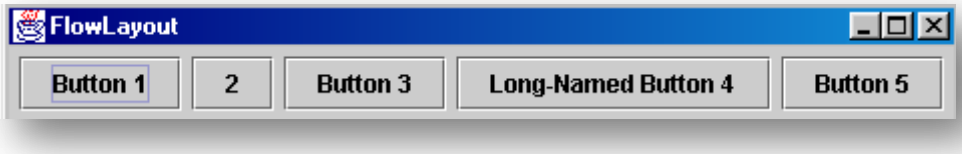

Las clases de Swing descienden de la clase “Container” de AWT.

La clase JComponent es la clase superior de la jerarquía de componentes Swing. Esta clase es una subclase de la clase java.awt.Container y, por tanto, es a la vez un componente y un contenedor en el sentido del AWT. En resumen todos los componentes de Swing descienden de java.awt.Container y java.awt.Component.

3. Añadimos los componentes simples al contenedor de alto nivel, dándoles un tamaño y posición dentro de éste.

Los paquetes java.awt y javax.swing proporcionan diversas clases que nos permiten especificar la disposición de los elementos en un contenedor. Estas clases se denominan Layout.
Hay varios tipos de Layout, que distribuyen los componentes en el contenedor de diversas maneras:

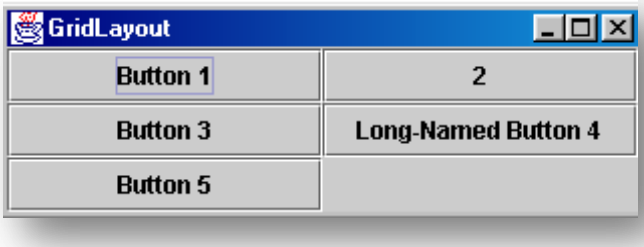
- FlowLayout: Incluye los elementos horizontalmente de forma secuencial




- BorderLayout: Divide el espacio en 4 bordes y el centro


- BoxLayout: Incluye los elementos verticalmente de forma secuencial


- GridLayout: Divide el espacio en una cuadrícula de nxm

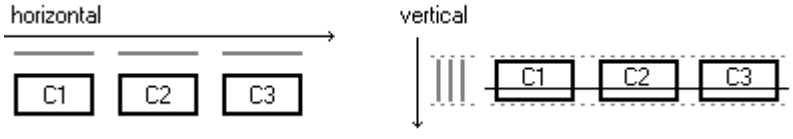



- GroupLayout:

  Es el Layout que he utilizado para esta interfaz ya que, a diferencia de los demás, trabaja con la dimensión vertical y horizontal por separado.

  De esta manera no tenemos que tener en cuenta ambas dimensiones a la vez, y el tipo de distribución, a la hora de colocar un componente en una determinada posición, basta con especificar la posición vertical y horizontal de cada componente dentro del contenedor.

  Esto es una ventaja, porque permite colocar componentes dentro de un contenedor de manera más intuitiva, y sin estar sujetos a una determinada distribución.



El Group Layout tiene una ventaja más, y es que puede manejarse de manera gráfica (sin necesidad de escribir el código manualmente) con un plugin que podemos integrar en el entorno de programación, Jigloo.

Gracias a esto, utilizamos la vista de la aplicación que ofrece Jigloo, situamos el componente en el lugar deseado, ajustamos su tamaño, y el código necesario para establecer el tamaño del componente, posición horizontal y posición vertical, se autogenerará.

## 4. Dotar de funcionalidad a los componentes

Tanto AWT como Swing tienen en común un sistema para gestionar los eventos que se producen al interactuar el usuario con la interfaz gráfica.

Para cada objeto de la interfaz gráfica, se pueden definir "objetos oyentes" (*Listener*) , que esperan a que suceda un determinado evento sobre la interfaz. Por ejemplo se puede crear un objeto oyente que esté a la espera de que el usuario pulse sobre un botón de la interfaz, y si esto sucede, él es avisado, ejecutando determinada acción.

En resumen, existen tres conceptos clave en un proceso de gestión de eventos:

- El objeto o componente de la interfaz (botón, campo de texto, etc.)
- La interfaz Listener (ActionListener, WindowListener, etc.) asociado a un objeto.
- El evento (hacer clic en un botón, pulsar una tecla, etc.)

Estos son algunos de los eventos más comunes y sus Listeners correspondientes:

| Acción que desemboca en el evento | Tipo de oyente |
|---|---|
| El usuario pulsa un botón o Enter mientras escribe en un campo de texto | ActionListener |
| El usuario cierra un marco (ventana principal) | WindowListener |
| El usuario pulsa un botón del ratón | MouseListener |
| El usuario mueve el ratón sobre un componente | MouseMotionListener |
| El componente se hace visible | ComponentListener |
| El componente obtiene el foco del teclado | FocusListener |
| La selección de la tabla o la lista cambia | ListSelectionListener |

El modelo de eventos de AWT depende del paquete *java.awt.event*, que en Swing se amplía con el paquete *javax.swing.event*.

(Los componentes Swing sólo se pueden usar con el modelo de eventos a partir del JDK 1.1. No admiten el modelo de eventos del JDK 1.0.)