RIELAC
Revista de Ingeniería Electrónica, Automática y Comunicaciones

Diseño e implementación de un software para equipos médicos modulares

Hernán D. Pérez Villanueva, Zuliet Medina Rodríguez, Adiel Alfonso Cordovi, José L. Casanova Pacheco, Luis R. Corrales Lay, Alberto S. Prieto Moreno

RESUMEN / ***ABSTRACT***

Este trabajo trata sobre el diseño e implementación de un software para equipos médicos modulares dedicados a la monitorización de pacientes en terapias. Para ello se utiliza como caso de estudio la familia de monitores de pacientes Doctus del fabricante cubano COMBIOMED. Se persigue el objetivo de generar un software lo suficientemente flexible que permita hacer modificaciones en su distribución, cambios en la configuración de los módulos que admite, y modificación de los algoritmos de procesamiento de señales digitales; sin tener que realizar modificaciones al software. Para el proceso de desarrollo se utiliza una metodología Agile modelo-V, y para la implementación se utiliza la tecnología .NET 6 de Microsoft.

Palabras claves: dispositivos médicos modulares, diseño de software, monitoreo de pacientes

This work is about the design and implementation of software for modular medical equipment dedicated to patient monitoring in therapies. The Doctus family of patient monitors from the Cuban manufacturer COMBIOMED is used as a case study. The objective is to generate software that is flexible enough to allow modifications in its distribution, changes in the configuration of the modules it supports, and modification of digital signal processing algorithms, without having to make any changes to the software. An Agile V-model methodology is used for the development process, and Microsoft's .NET 6 technology is used for the implementation.

Keywords: modular medical devices, software design, patient monitoring

Design and Implementation of a software for modular medical devices.

1. -INTRODUCCIÓN

El desarrollo de software para dispositivos médicos es un proceso complejo debido a las regulaciones sobre el sector. Al contrario de otros dispositivos inteligentes, los diseñadores de dispositivos médicos deben ser cautelosos con la implementación de cambios, debido a la estabilidad de largo plazo y durabilidad de los equipos médicos. Los usuarios suelen desarrollar hábitos cognitivos y expectativas de formas típicas de interacción con los equipos [1]. Las regulaciones llegan al punto en el que no solo se norma la validación del producto, sino también el proceso de desarrollo de todo software involucrado en él. El proceso en sí mismo de la verificación y validación representa un reto para los desarrolladores, dado que puede llegar a ser abrumador y complejo si solo se toma la perspectiva general de todas las variables a considerar [2]. La tendencia actual es a implementar un enfoque Agile con modelo V para el desarrollo desde hace más de una década, conforme a los marcos regulatorios más comunes [3]. En [4] se presenta un estudio de las metodologías basadas en este modelo en la literatura y se propone un modelo híbrido que combina un desarrollo dirigido por planificación con el modelo V, dando lugar a un Modelo-V Agile Mejorado (EAV por sus siglas en inglés). En este enfoque se resuelven algunas de las dificultades presentes en el Modelo V, como los riesgos que introduce el diseño de la arquitectura de manera agile, dado que en este enfoque esta responsabilidad queda relegada a la parte de planificación. Este se basa en una etapa inicial de planificación y luego un desarrollo Agile siguiendo dicha planificación, cumpliendo con los requerimientos del modelo V donde en cada etapa del desarrollo corresponden pruebas de verificación de cada parte del diseño de software, finalizando con la validación de los requisitos de usuario. Durante el desarrollo del software del caso de estudio se utiliza este enfoque.

Recibido: 09/2024 Aceptado: 11/2024

El caso de estudio desarrollado es un monitor de pacientes de la familia Doctus, del fabricante cubano COMBIOMED. Para este caso se estudia cómo realizar un software que sea flexible ante los cambios y que permita su evolución sin necesidad de modificaciones a este, de manera que se minimicen los esfuerzos de mantenimiento y desarrollo. Otra necesidad del fabricante es la capacidad de dar mantenimiento los equipos con afectaciones reduciendo el tiempo que estos se encuentran sin prestar servicio. Con el propósito de mejorar este servicio se utilizan los diseños modulares, de forma que, si la rotura se origina en un módulo particular del equipo, basta con remplazarlo para que el equipo vuelva a ser operativo. Por lo que el software desarrollado por este trabajo debe soportar el reconocimiento e integración de un hardware modular. Esto implica que la disponibilidad de sus funcionalidades dependerá de los módulos presentes en el equipo.

La tendencia actual en cuanto a equipos de monitoreo de señales vitales es a utilizar dispositivos remotos e implantes como una medida para determinar problemas antes de que ocurran o para el seguimiento de pacientes una vez salen del hospital [5]. Sin embargo, mientras el paciente se encuentra hospitalizado, se mantiene con un equipo de monitoreo en el salón, pero estos equipos tienen cada vez más capacidades de conectarse a unidades centrales para mantener un registro y procesar las señales vitales de cada paciente, acorde a la filosofía IoT [6]. Existen algunos enfoques, como el diseñado en [7], donde se propone un sistema de monitoreo a partir de varios sensores inteligentes conectados al teléfono móvil del paciente o doctor. Estos dispositivos se encargan de presentar de forma visual la información y transmitirla hacia los servidores en internet. Si se busca un enfoque modular, donde los componentes con errores pueden ser sustituidos sin afectar al resto del sistema, esta puede ser una buena propuesta, sin embargo, también tiene sus desventajas. Al ser un sistema portable, si no se encuentra cerca del teléfono móvil o si este está desconectado de internet, no es posible detectar posibles situaciones críticas para el paciente.

Una idea similar es desarrollada en [8], donde en lugar de usar un teléfono móvil para la identificación y procesamiento de los sensores, se utiliza un procesador dedicado conectado a los sensores. Dicho controlador envía los datos a un servidor en internet desde donde pueden ser visualizados usando cualquier dispositivo. Este enfoque libera la dependencia del teléfono móvil como procesador, pero también requiere de una conexión a internet para funcionar, además de que este sistema podría implicar demoras importantes en la notificación de un evento anómalo en los signos vitales. Estos problemas son identificados en [9], donde los autores realizan una encuesta con el fin de identificar las deficiencias en sistemas de monitoreo remoto. Los elementos más relevantes detectados fueron la pobre conectividad, la sobrecarga de alarmas sin importancia (posiblemente opacando otras alarmas más críticas) y las enormes cantidades de datos que deben ser procesadas con prontitud para detectar situaciones anómales en los pacientes. La solución a estos problemas radica en tener una buena infraestructura de comunicación y procesamiento de los datos, así como algoritmos eficientes con respecto al tiempo y precisos en la detección de eventos anómalos en el paciente. En Cuba estos sistemas están aún en una etapa inicial, la mayoría de los hospitales no cuentan con la infraestructura necesaria para desplegar sistemas basados en IoT, por lo que se decide dejar estas características para futuras versiones de la aplicación.

Una propuesta similar a los equipos clásicos de monitoreo de terapia es presentada en [10], donde se emplea un microcontrolador Arduino para el reconocimiento de distintos módulos y sensores de medición de signos vitales. Luego la información recolectada es enviada a una Raspberry Pi para su visualización y reenvío a servidores en la nube. En estos servidores los datos son analizados y cualquier eventualidad o alarma es comunicada al personal médico en forma de SMS. Este prototipo carece de funcionalidades importantes de un monitor de paciente como la notificación de las alarmas en el propio dispositivo, o la configuración de los parámetros límites para estas alarmas.

La idea de diseñar un equipo modular, implica el diseño de una comunicación entre los distintos módulos. En [11] se realiza un estudio de las principales tecnologías de comunicación en dispositivos médicos, desde sensores hasta sus procesadores. Para la comunicación interna entre dispositivos, identifican dos grupos según su aplicación: los equipos remotos y los equipos conectados por cables. Para los segundos, se utilizan las tecnologías receptor-transmisor universal asíncrono (UART), circuito integrado (I2C) e interfaz periférica serial (SPI).

El objetivo de este trabajo es conseguir el diseño y desarrollo de un software que mantenga las funcionalidades de los equipos de monitorización de pacientes desarrollados en la empresa COMBIOMED, brindando una solución que pueda ser utilizada en distintos equipos de monitorización solo con la configuración del software, sin la necesidad de realizar cambios sobre él. El fabricante planea comenzar la implementación de diseños modulares en sus equipos por las ventajas que estos aportan, por lo que el software debe ser capaz de reconocer e identificar los distintos módulos de cada equipo. Para esto se diseñó un protocolo de comunicación entre el software y los módulos. Se debe definir la estructura de configuración de los módulos y su visualización en pantalla para poder aplicar un tratamiento uniforme a cada uno de ellos. Los algoritmos para la detección de eventos cardiacos o anomalías en la respiración se están constantemente mejorando con el desarrollo científico en esta rama. En la última década, con el uso de la inteligencia artificial, han surgido diversas técnicas que ofrecen excelentes resultados [12]. Si el software será diseñado para ser utilizado en varias versiones de varios equipos, es necesario suministrar una vía para modificar los algoritmos utilizados sin que esto implique la modificación del software de monitoreo. Otro objetivo es conseguir adaptar la interfaz con la que están familiarizados los usuarios de esta familia de equipos a un diseño que permita

un tratamiento uniforme de cada módulo. Contrario a las versiones actual y anteriores, donde cada parámetro fisiológico del paciente tiene funcionalidades diferentes y por tanto, su visualización tiene particularidades para cada uno. Para llevar a cabo esta restructuración del diseño de manera efectiva se consideran las conclusiones obtenidas en [13], donde se realiza un estudio de las prácticas en las interfaces de usuario de equipos médicos que son efectivas o contraproducentes. De estas resaltan el tamaño de la información con respecto al de la pantalla y el uso de distintas jerarquías de mensajes según su relevancia. También se tienen en cuenta las directrices del personal médico asociado al proyecto, para establecer que zonas de la pantalla tienen mayor criticidad en el seguimiento al paciente.

Debido a la carga de configuración que acarrea dar soporte a un software tan flexible, también es objetivo del trabajo diseñar una vía de configuración que sea fácil y segura para el fabricante. De manera que pueda almacenar las configuraciones previamente establecidas para cada uno de sus productos e instalarlas en la línea de producción o de forma remota, en casos en los que se desee actualizar la configuración de un equipo en uso sin la necesidad de retirarlo de su ubicación.

En las siguientes secciones se presenta el diseño e implementación de una solución para satisfacer los objetivos planteados. La sección 2 define los pasos propuestos para la realización del análisis y diseño de la solución. En la sección 3 se presenta el diseño hardware propuesto para la solución. Se propone un protocolo de comunicación serial para esta configuración hardware y como este se utiliza en el descubrimiento de distintos módulos del equipo. En la sección 4 se definen las estructuras que conforman cada módulo y se explica su origen a partir de las versiones anteriores de esta familia de equipos médicos. La sección 5 está dedicada a explicar los requerimientos de procesamiento de señales fisiológicas y las estructuras de datos y de procesamiento que se proponen para satisfacerlos. La sección 6 explica como la solución permite manejar el volumen de configuración definido para cada módulo a través de una aplicación externa, a disposición del fabricante del equipo. La sección 7 describe los patrones y buenas prácticas seguidas durante el desarrollo del software, así como las tecnologías y plataformas empleadas.

2.- Propuesta de solución

Los objetivos de este artículo requieren el diseño de una solución para un software configurable que soporte un hardware modular, manteniendo las funcionalidades de un software anterior para un hardware monolítico. Conseguir esta meta requiere de un análisis de las funcionalidades de los softwares de equipos anteriores y de las que se deseen agregar para la nueva versión, con el propósito de determinar cómo se adoptarían cada una en el nuevo software. Es importante definir el tratamiento que se le dará a cada funcionalidad, debido a que en un software que atiende equipos modulares debería ser capaz de darle un tratamiento estándar de acuerdo a la configuración de cada uno, sin tener que conocer particularidades de un módulo. Para la realización de este análisis se proponen los siguientes pasos:

1. Identificar las funcionalidades de los equipos anteriores y las funcionalidades que se deseen añadir en la nueva versión.
2. Identificar los módulos del equipo.
3. Determinar que funcionalidades del equipo son generales (independientes sus módulos) y cuales están asociadas a los módulos. Las funcionalidades que son particulares del equipo no necesitan ser configuradas en cada módulo, ya que serán parte del dispositivo independientemente del módulo que se encuentre disponible.
4. De las funcionalidades asociadas a los módulos, identificar cuáles son comunes, cuales están presentes en varios, pero no en todos y cuales son únicas de un módulo. A partir de esta identificación, se puede determinar:
    - Las funcionalidades comunes para todos los módulos conformarán configuraciones requeridas para estos.
    - Las que pueden estar presentes o no, serán configuraciones opcionales de los módulos.
5. Para las funcionalidades que se identifiquen como únicas de un módulo específico se debe diseñar mecanismos de abstracción que permita configurarlas a sus módulos correspondientes, pero que puedan ser ejecutadas sin que el software tenga que conocer los detalles del funcionamiento de cada módulo.
6. Diseñar una estrategia de configuración teniendo en cuenta los casos presentes de los antes mencionados.

Como parte de la primera etapa se realiza una captura de requisitos de usuario en la que se identificaron 71 de ellos. A partir de ellos, se identificaron los siguientes módulos: Electrocardiograma, Presión invasiva, Presión no invasiva, Oximetría, Capnografía y Relajación muscular. De los requisitos capturados, 58 se son funcionalidades asociadas a los módulos del dispositivo. En las siguientes secciones se dará continuidad a los siguientes puntos del análisis.

3.- DISEÑO HARDWARE DEL DISPOSITIVO

El caso de estudio sigue un diseño de hardware común entre dispositivos médicos de monitoreo: varios microcontroladores para obtener las mediciones, y otro más potente dedicado al procesamiento de las mediciones, visualización de la información, y la interacción con el usuario. Para mayor claridad, en este trabajo nos referiremos a estas dos capas de hardware como capa de medición y capa de visualización. Aunque estos términos son una simplificación de sus responsabilidades, refleja el requisito más crítico que deben satisfacer. Esto permite una separación de responsabilidades en la que el controlador de visualización recibe las muestras de las variables de interés, abstrayéndose de su principio de medición y los componentes electrónicos que se utilizan para su procesamiento. Pero también implica que es necesario establecer un protocolo de comunicación entre estas dos capas. La figura 1 ilustra este diseño.

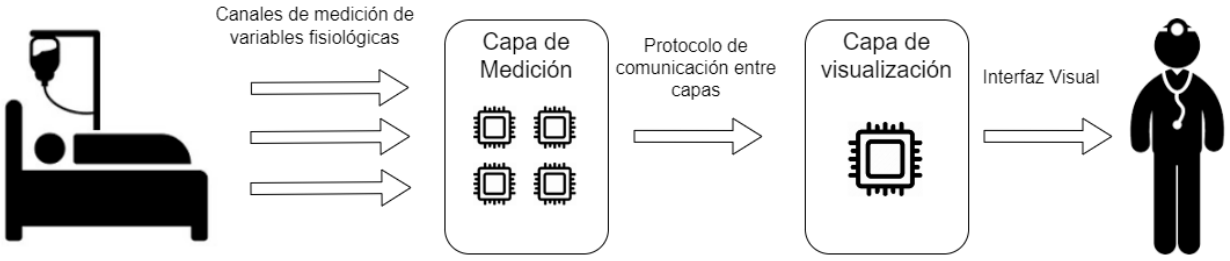

Figura 1

Diseño hardware del equipo.

3.1- PROTOCOLO DE COMUNICACIÓN ENTRE CAPAS

Un requerimiento funcional esencial para estos equipos es la velocidad de comunicación. Muchas de las señales fisiológicas que se obtienen del paciente tienen un período de muestreo en el orden de los milisegundos, y se necesita esta alta densidad de muestras para un buen rendimiento de los algoritmos que las procesan en la capa de visualización. Los algoritmos de procesamiento se implementan en esta capa debido a sus mayores capacidades de procesamiento, una fundamentación más detallada de esta decisión se explica en la sección 4. Teniendo esto en cuenta se decide usar un protocolo basado en mensajes sobre protocolo serial. El protocolo serial es flexible en cuanto al ajuste de las velocidades de comunicación y permite diseñar byte a byte los mensajes de comunicación, lo que permite diseñar un protocolo eficiente para las necesidades del equipo, con el mínimo posible de bytes por mensajes. Considerando estas necesidades se diseñó una estructura base para todos los mensajes del protocolo, ilustrada en la figura 2.

AAh	55h	Cantidad de Bytes	Byte de cabecera	$Byte_0$	$Byte_1$	...	$Byte_n$	Byte de comprobación

Figura 2

Estructura de un mensaje en el protocolo de comunicación entre capas.

Los mensajes iniciarán con dos bytes de un valor establecido, los cuales se emplearán por los algoritmos de lectura para determinar el inicio de un mensaje. Se utilizan dos bytes, y no solo uno, para reducir las probabilidades de que se interprete un inicio de mensaje de forma errónea. Seguido al inicio se envía en otro byte la cantidad de bytes del mensaje a partir de esta posición, esto será empleado por el algoritmo de lectura para reconocer si un mensaje está terminado o aún faltan bytes por recibir. Al usar solo un byte para esta función estamos limitando a que un mensaje puede contener hasta 126 bytes de datos más los 3 de la estructura del protocolo, ya que es el mayor valor que se podría enviar en este byte de cantidad. Se evaluó esta limitante con las necesidades de la solución y se determinó que esta capacidad de información satisface incluso los mensajes con mayor densidad de información. Le sigue un byte llamado byte de cabecera que identifica el tipo del mensaje. Algunos mensajes requieren la transmisión de datos adicionales, a esta necesidad se corresponden los bytes que le siguen al byte de cabecera, la cantidad depende de la naturaleza del mensaje. Finalmente se usa un byte de comprobación para el chequeo de integridad del mensaje. Estos equipos de monitoreo pueden utilizarse en quirófanos donde, debido al uso de bisturís eléctricos, puede haber un ruido eléctrico considerable. Este byte es una medida de seguridad ante esta situación. El algoritmo que se decide emplear para determinar este byte es el chequeo de suma de 8 bits. Este se basa en sumar todos

los bytes del mensaje y utilizar el byte menos significativo del resultado como chequeo, el receptor realiza la misma operación y comprueba que el resultado sea igual al recibido.
El comportamiento más común sería descartar los mensajes que no cumplan el chequeo, pero hay algunos casos cuya repercusión es demasiado importante como para descartarlos en caso de fallos. Para evitar estas situaciones, a estos mensajes se les establece un mensaje de confirmación. El mensaje de confirmación tiene como dato la cabecera del mensaje que se quiere confirmar. Si un mensaje requiere confirmación, el emisor esperará un tiempo válido para que el receptor confirme el mensaje, en caso de superarse el tiempo, el emisor reenviará el mensaje. Luego de una cantidad determinada de reenvíos fallidos el sistema detectará un error de comunicación y se lo comunicará al usuario. Esta es un error fatal para el equipo, que requerirá de revisión técnica.
A continuación, se listan los principales tipos de mensajes y si requieren de esta confirmación.

- Medición: contiene las muestras capturadas en un módulo. No requiere confirmación. Es el mensaje que se transmite con mayor frecuencia. Las muestras de la medición se envían en los bytes de datos.
- Reconocimiento de módulos: informa a la capa de visualización de los módulos activos en el hardware. No requiere confirmación. Solo se envía una vez al iniciar el equipo.
- Confirmación: mensaje empleado para confirmar otros mensajes que lo requieran.
- Estado de batería: transmite información asociada al estado de la batería del equipo. No requiere confirmación.
- Alarma: transmite estados de alarma detectadas en la capa de medición hacia la capa de visualización.
- Configuración: transmite ajustes de parámetros modificados en la capa de visualización que implican cambios en el comportamiento de la capa de medición. Requiere confirmación.
- Comando: mensaje que puede ser enviado en ambos sentidos. Implica una solicitud del emisor para la ejecución de una tarea en el receptor. Requiere confirmación. Se le asocia un byte de datos que identifica el tipo de comando a emplear. Algunos de los comandos definidos:
    - Solicitud de módulos.
    - Solicitud de medición.
    - Calibración de módulo.
    - Cambio de canal de medición.
    - Solicitud de apagado.
- Botón: mensaje transmitido cada vez que se detecta la activación de algún elemento de la interfaz mecánica con el usuario (botones y codificadores digitales). En sus bytes de datos se incluye el identificador del elemento activado. No requiere confirmación.
- Disparador: mensaje transmitido para desencadenar alguna acción en la capa visual pero cuyo detonante fue detectado en la capa de medición. Por ejemplo: la detección de marcapasos en el paciente. No requiere confirmación.
- Estados: mensaje transmitido para mostrar algún estado que afecta el funcionamiento correcto del equipo. Por ejemplo, la detección de electrodos sueltos en la medición de ECG. En los bytes de datos se envía un identificador que está asociada a un mensaje a mostrar en la capa de visualización. No requiere confirmación.

3.2- DISEÑO MODULAR

La solución está diseñada para que funcione de forma modular, las funciones del hardware estarán distribuidas en distintos módulos (tarjetas electrónicas específicas para cada función). El software de la capa de visualización debe tener la capacidad de reconocer cada módulo de hardware, comunicarse de forma independiente con él y generar su contenido visual a partir de los módulos que haya reconocido. El diseño modular de equipos médicos ofrece varias ventajas:

- Reduce el costo de adquisición al solicitar solo los módulos requeridos por cada servicio hospitalario.
- Mejora su aprovechamiento al permitir la sustitución de módulos dañados en lugar de la reposición de todo el equipo o solicitar reparación al fabricante.
- Permite al cliente adquirir módulos de repuesto cuya instalación puede ser realizada por personal técnico del hospital, mejorando los tiempos de disponibilidad de los dispositivos.
- Permite al fabricante la mejora del equipo mediante el desarrollo de nuevos módulos, sin necesidad de modificar el software principal del equipo, lo que implica que los nuevos módulos serán compatibles con las unidades ya comercializadas.

Usar un diseño modular junto al diseño de hardware antes mencionado añade una complejidad al sistema: la capa de visualización debe reconocer la presencia de módulos de forma dinámica y modificar su comportamiento en consecuencia.

En el hardware de los módulos pueden encontrarse dos escenarios: módulos que tienen un controlador dedicado y módulos que usan un controlador compartido con otros. Esta distribución se realiza buscando la optimización de los recursos de hardware. Para uniformar estos dos escenarios se introduce el concepto de **concentrador de módulos**, refiriéndose al controlador que se comunica con la capa de visualización y que puede agrupar uno o más módulos. También pueden existir controladores cuyo propósito no está asociado a ningún módulo en particular, si no que cumplen funciones del hardware del equipo, como pueden ser la detección del accionamiento de los botones o el control de impresoras presentes en algunos modelos. Estos últimos siempre estarán presentes en el equipo, pero la cantidad puede variar de un modelo a otro, por lo que un reconocimiento dinámico puede ser útil para lograr un software que pueda usarse en diferentes modelos.

Ante este requerimiento se decide establecer un protocolo de descubrimiento de módulos y otros componentes de hardware. Como una restricción natural es que cada uno de estos concentradores debe comunicarse con la capa de visualización por un puerto serie diferente. Por lo que el protocolo comienza con la identificación de los puertos series accesibles. Luego se envía a cada uno de estos puertos un mensaje de solicitud de identificación de tipo de dispositivo. Si el puerto tiene conectado un componente de la capa de medición debe responder con una identificación de su responsabilidad: concentrador de módulos, impresora, botones de navegación, por citar algunos ejemplos. En el caso particular de los concentradores, se les solicita que identifiquen los módulos que manejan, como se ilustra en la figura 3. Esto implica que la capa de visualización tiene que conocer de antemano el identificador de cada posible módulo que se puede añadir al dispositivo. Si un puerto serie no responde el mensaje de identificación en un tiempo determinado, se asume que no es un componente válido del dispositivo.

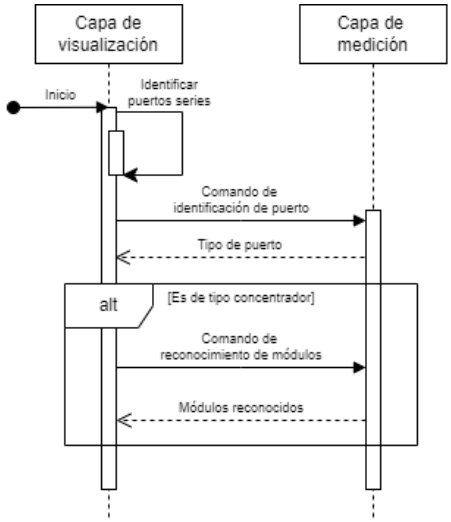

Figura 3

Diagrama de secuencia del protocolo de reconocimiento de módulos.

4.- ESTRUCTURA DE MÓDULOS

Los equipos de monitoreo suelen contar con una agrupación lógica de los datos que se muestran, basados en la naturaleza fisiológica de cada uno. Por ejemplo, un monitor de paciente contiene datos asociados a: electrocardiogramas (ECG), respiración, oximetría, capnografía, entre otros parámetros. Lo cual nos lleva a una delimitación de los módulos de forma natural. Pero existen particularidades que complejizan esta idea: los electrodos utilizados para la medición de electrocardiograma, también se utilizan para obtener la señal de la respiración del paciente.

Atendiendo que un módulo cuenta con el hardware asociado a las mediciones de las variables fisiológicas, no es correcto delimitar que respiración y electrocardiograma son diferentes módulos, sin embargo, sigue siendo importante mantener un agrupamiento para su visualización y configuración. Ante este problema se decide establecer el concepto de parámetro fisiológico, como un agrupamiento de variables, su visualización, su procesamiento y su configuración dentro de un mismo módulo hardware. Por lo tanto, un módulo puede contener uno o más parámetros fisiológicos.

La información asociada a las señales fisiológicas de un paciente se presenta en dos formatos: gráficos de líneas para señales continuas, cuya utilidad está en el análisis del comportamiento temporal de la señal; y campos alfanuméricos para visualizar indicadores discretos, cuya utilidad radica en el estado actual de dicha variable. Se decide utilizar estas dos distribuciones para la visualización de la información asociada a un parámetro fisiológico. En el campo alfanumérico, lo más común es presentar valores discretos, pero también se utilizan para mostrar íconos, los principios de medición asociados al parámetro y el estado actual de la medición (en caso de que las mediciones no sean continuas). Incluso, los propios valores discretos pueden tener distintas prioridades según su importancia, lo cual afecta su tamaño en pantalla.

A esta variedad de formas de presentar la información se les denominó indicadores, y pueden ser de diversos tipos en función del tipo de información o de su formato de presentación. Los indicadores numéricos que requieren mayor atención del personal médico tienen mayor tamaño, mientras que los de menor relevancia tienen menor tamaño. Para distinguirlos se les clasificó como indicadores numéricos principales y secundarios, respectivamente.

En el caso de ECG, el campo alfanumérico presenta un ícono de corazón parpadeante sincronizado con los latidos del paciente. Para soportar esta característica se decide definir también que un campo alfanumérico puede presentar varios indicadores de íconos. Otra funcionalidad particular es en el caso de la presión no invasiva, que, a diferencias de otros parámetros, no se mide de forma permanente, sino que su medición es desencadenada por el usuario. Esta medición puede responder a varios patrones, desde una medición individual hasta programar mediciones periódicas por un tiempo determinado. Todo este comportamiento se visualiza en el campo alfanumérico, por lo que se hace necesario definir un cuarto indicador para los intervalos de medición. Cada uno de estos formatos de visualización se asocian a variables fisiológicas manejadas por el módulo al que pertenecen, ya que ellos presentan el valor actual de estas variables. En la figura 4 se ilustra las relaciones entre los conceptos explicados.

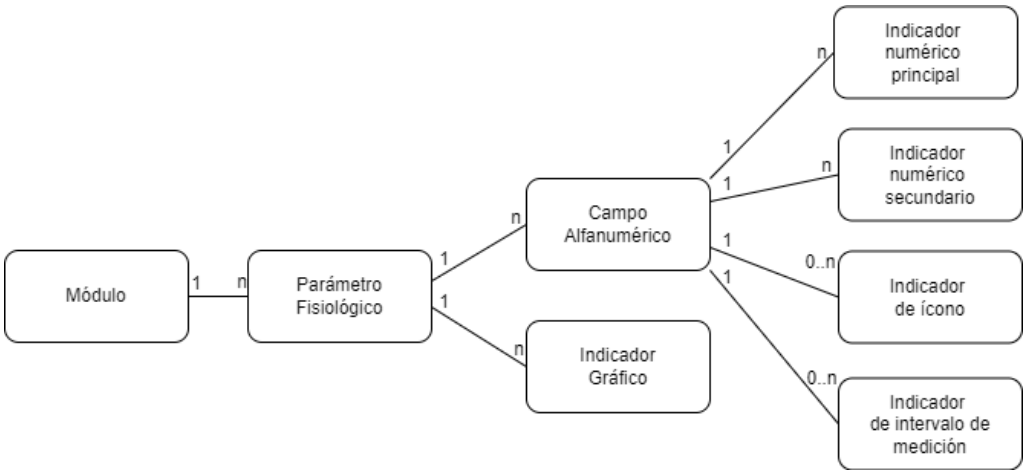

Figura 4

Estructura de configuración de un módulo.

La definición de estas estructuras fue resultado de un análisis de otras versiones de la familia del equipo caso de estudio, otros equipos del propio fabricante y un estudio de las tendencias en los principales fabricantes de equipos médicos de monitorización [14, 15]. El objetivo es definir una configuración que permitiera describir al software cómo presentar la información asociada a cada módulo, de forma que sea flexible para su utilización en diferentes modelos de dispositivos. Este trabajo busca definir como se utiliza un enfoque de estandarización para poder configurar módulos que pueden llegar a tener comportamientos y presentaciones diferentes.

4.1- ESTRUCTURA DE INDICADORES GRÁFICOS

Los indicadores gráficos son útiles para los médicos en la detección de anomalías en las variables fisiológicas de un paciente a partir del comportamiento temporal de estas. Según el comportamiento de la señal de electrocardiograma los doctores pueden diagnosticar eventos anómalos, como algunos tipos de arritmias cardiacas. La figura 5 ilustra esta funcionalidad. El diseño visual de estos indicadores consiste en gráficos de líneas en tiempo real que siguen el comportamiento de las mediciones realizadas por el hardware sobre el paciente. Para mantener un flujo continuo, al alcanzar el final de la pantalla, los gráficos

comenzarán a generarse desde el inicio, usando una pequeña franja vacía para ayudar a ubicar las nuevas mediciones. Estos gráficos se consideran en tiempo real porque tienen una actualización constante del valor actual de la medición, con un retardo implícito en la realización de la medición, la transmisión hacia la capa de visualización y el procesamiento de los valores. Es crucial que los gráficos respondan al estado actual del paciente y que su tiempo de retardo sea mínimo, ya que el tiempo de respuesta de un médico hacia una anomalía de un signo vital es esencial a la hora de prevenir daños en el paciente. Esto se tuvo en cuenta a la hora de desarrollar el software, buscando la mayor eficiencia en las etapas de comunicación y procesamiento de las señales. Los datos que son destinados para esta visualización tienen un flujo más directo en la aplicación, se sacrifican el uso de técnicas que contribuyen a la legibilidad y la separación de responsabilidades en el código, con el fin de reducir el tiempo de respuesta.

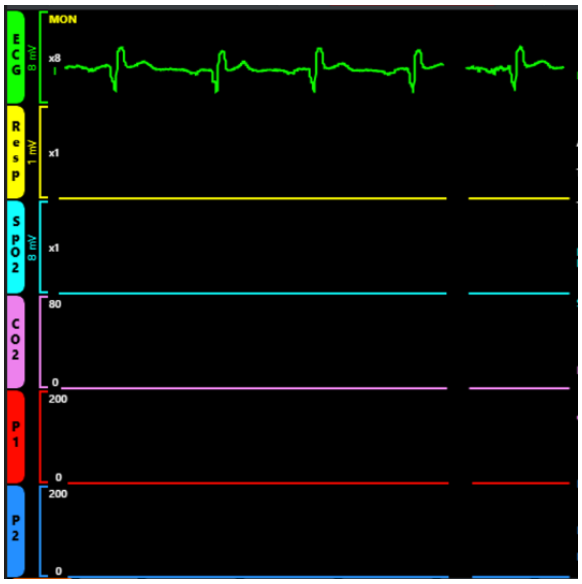

Figura 5

Indicadores gráficos de la solución.

Otra complejidad presente en esta característica es la necesidad de mantener sincronizadas las señales fisiológicas. Las muestras provienen de distintos módulos de hardware, que las obtienen con diferentes frecuencias de muestreo. Los módulos no se encuentran necesariamente sincronizados, por lo que tampoco enviarán sus muestras con la misma frecuencia. Esto implica que en un mismo periodo de tiempo se puede tener distinta densidad de muestras para cada variable. Para mantener un flujo sincronizado de los gráficos de líneas es necesario establecer un período de tiempo, de forma que cada vez que transcurra el gráfico actualizará sus valores. La elección de este valor afectará la fluidez con la que se visualiza el graficado. A este período de tiempo se le llama período de refrescamiento. Si el período de muestreo de una variable es inferior al período de refrescamiento, se tendrán varias muestras por cada actualización del gráfico. Ante esta situación se pueden valorar varios enfoques: tomar solo la última muestra (refleja el estado más actual de la variable), promediar (refleja el comportamiento medio de la variable), usar el mínimo o el máximo (refleja los comportamientos extremos de la variable). Pero dado que para los doctores es muy útil la topología de la señal con la mayor resolución posible, se decide que se graficarán todas las muestras, pero ocupando el mismo espacio en el gráfico, el espacio correspondiente al tiempo transcurrido en el período de refrescamiento. Este comportamiento es reflejado en la figura 6. En el caso de que una señal tenga un periodo de muestreo superior al periodo de refrescamiento, eventualmente ocurrirán ciclos donde no se tenga ninguna nueva muestra disponible para dicha variable. En estos casos, se tomará para graficar el último valor medido de la variable.

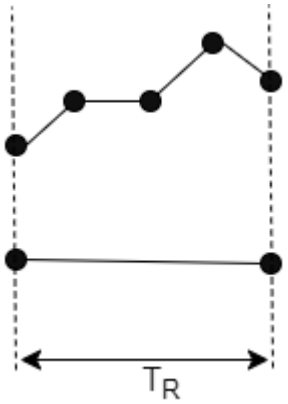

Figura 6

Visualización de señales fisiológicas de distintas densidades.

4.2- ESTRUCTURA DE CAMPOS ALFANUMÉRICOS

Los equipos actuales de la familia Doctus son el resultado de mejoras en el transcurso de los años, implementadas por diferentes grupos de desarrollo. Como resultado, tenemos campos alfanuméricos con diferentes formatos y distribución de la información. Esto no cumple con la uniformidad necesaria para un software modular, ya que se necesitaría tener conocimiento de la forma particular de visualizar cada parámetro. En la figura 7 se puede visualizar la distribución del contenido de los campos alfanuméricos en el último modelo de la familia Doctus: el monitor de paciente Doctus VIII. Una solución a este problema sería permitir que el formato de visualización de los campos sea configurable para cada parámetro, pero esto implica una mayor complejidad en la configuración del equipo. Por lo que se decide reducir el nivel de personalización estableciendo una distribución general de los distintos formatos, otorgar a cada indicador una zona específica del campo alfanumérico, y permitir la personalización del comportamiento del indicador, pero no del campo alfanumérico.

Figura 7

Campos alfanuméricos del equipo Doctus VIII.

Luego de clasificar los diferentes formatos en los que se presenta la información en un campo alfanumérico, se define una estructura general que sea aplicable a cada parámetro, teniendo en cuenta que la presencia algunos de los formatos para presentar la información pueden no estar presentes. Con la guía de diseñadores y doctores de la empresa COMBIOMED, se diseñó una generalización de los campos alfanuméricos, siguiendo la distribución mostrada en la figura 8.

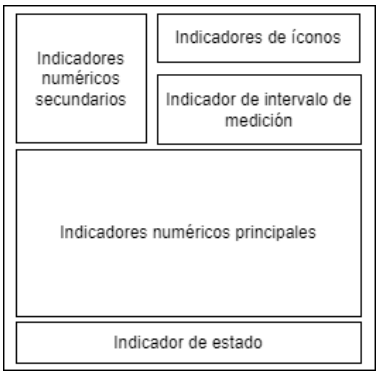

Figura 8

Estructura de campos alfanuméricos de la familia Doctus.

Para un mayor aprovechamiento del espacio en pantalla se decide dar una proporción cuadrada al campo, contraria a la rectangular que presentaba anteriormente. La zona con mayor espacio es la de los indicadores numéricos principales, debido a que son los de mayor prioridad de visualización. Dado que en algunas variantes no se utilizan íconos o indicadores numéricos, estás zonas ocuparán el espacio designado solo en caso de ser utilizados, de lo contrario, este espacio será utilizado por las zonas adyacentes. En el Doctus VIII los campos alfanuméricos están distribuidos en una sola columna, pero con la nueva proporción cuadrada, no es posible colocar la misma cantidad de campos. Ante este problema se decide distribuirlos en varias columnas según la cantidad de campos que hayan sido configurados y que se encuentren visibles (en la aplicación es posible elegir que campos se visualizarán y cuales se ocultarán). Esta estructura se aplicó a los campos alfanuméricos dando como resultado la figura 9, donde se observa una mayor uniformidad en la distribución de la información sin dejar de cumplir con las necesidades particulares de cada campo. En la figura 10 se ilustra la distribución en pantalla de estos campos según la cantidad de ellos que estén visibles, ya que el usuario puede configurar que algunos campos estén habilitados y otros no.

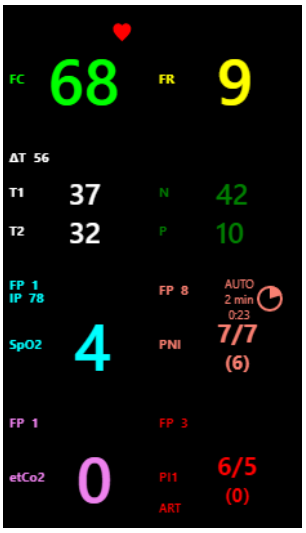

Figura 9

Resultado de aplicar la estructura general de campos alfanuméricos.

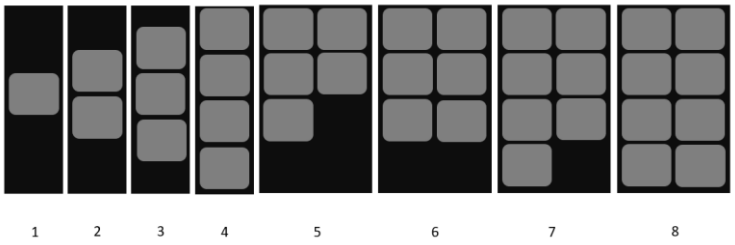

Figura 10

Distribución de campos alfanuméricos según la cantidad que haya visibles.

Si se observa el campo alfanumérico de la presión no invasiva (PNI) se nota que el formato en que se presenta la información es diferente al resto. En lugar de mostrar el valor de una variable, se muestran tres variables independientes: la presión diastólica, la presión sistólica y la presión media. Para soportar esta funcionalidad, se permite que un indicador numérico principal tenga varias variables asociadas a él, pero debe definir el formato en que se mostrarán. Por ejemplo, en el caso de la presión no invasiva se utilizaría el siguiente formato: "{0}/{1}\n({2})". Las llaves se utilizan para delimitar el índice de la variable cuyo valor se muestra en esa ubicación. Se utiliza el carácter de salto de línea para separar distintas líneas en el formato. Para alcanzar esta flexibilidad, que permite la presentación de los valores de múltiples variables dentro de un mismo indicador numérico principal, se implementó una funcionalidad que permite la interpretación de formatos escritos como texto. Donde, adicionalmente, se soporta el establecimiento de distintos tamaños de las letras para cada valor.

4.3- CONFIGURACIÓN DE MÓDULOS

Las estructuras definidas hasta este punto corresponden a requerimientos asociados a módulos que están presente en todos o en algunos de ellos. Correspondiente a la cuarta etapa del análisis propuesto. Si se intentara aplicar la filosofía de separación de responsabilidades a estos dispositivos, el software de un equipo modular ideal sería completamente agnóstico de los módulos con los que está trabajando y de su configuración, siendo solo su responsabilidad el control del flujo de información entre la capa de medición y su presentación e interacción con los usuarios. Pero, en la práctica, esto es difícil de conseguir. Debido a las limitaciones de los controladores de la capa de medición, no es factible que estos suministren la configuración que conlleva cada módulo durante el proceso de descubrimiento inicial. Como resultado, el procesador de visualización (que por sus propias responsabilidades debe ser más potente) debe contener la configuración antes mencionada de cada módulo que podría integrarse a su sistema. Por lo que el software puede dar un tratamiento uniforme a cada módulo, sin necesidad de realizar implementaciones dirigidas a módulos en específico. Esto añade la limitante de que la cantidad de módulos soportados por el equipo estará definida dentro de la configuración de su software, por lo que el soporte a nuevos módulos, implicaría una actualización de dicha configuración. En la sección 6 se presenta un mecanismo de configuración para minimizar esta limitante.

5.- PROCESAMIENTO DE SEÑALES FISIOLÓGICAS

Mantener la abstracción con respecto a los detalles específicos de cada módulo es el mayor desafío de un software modular, sobre todo cuando hay tanta variedad de comportamientos entre ellos. En la sección anterior se presentó una solución basada en la estandarización de las particularidades de cada módulo en conceptos comunes para todos. Sin embargo, se presentan casos en los que este enfoque no es posible, como lo es el procesamiento de las señales fisiológicas. Por ejemplo, para detectar eventos de arritmia o para calcular la frecuencia cardiaca es necesario aplicar algoritmos específicos para señales del módulo de electrocardiograma. Ante este problema un primer enfoque fue intentar delegar la responsabilidad de ejecutar estos algoritmos a los controladores de cada módulo, y que estos trasmitieran los valores resultantes del procesamiento de las señales medidas. Sin embargo, por las capacidades de procesamiento necesarias, implicaría usar tecnologías más costosas y complejas. Por este motivo se decide descartar este enfoque.

Dado que la única variante viable es el procesamiento de las señales fisiológicas en la capa de visualización, se decide diseñar una forma de aislarlo de la lógica de aplicación. La variante que se decide emplear es usar una librería externa donde se

definan los algoritmos de procesamiento. La aplicación del dispositivo deberá cargar de forma dinámica de esta librería y hacer un reconocimiento de estos. Se identificaron cuatro requerimientos que aplican a esta funcionalidad:

1. cálculo de variables,
2. filtrado de señales continuas para eliminar ruido eléctrico,
3. detección de eventos fisiológicos y
4. obtención de señales de referencia para otros algoritmos.

En el caso de los primeros tres, se estarán evaluando constantemente según se reciban las muestras requeridas, mientras que el último solo se ejecutará cada cierto tiempo o bajo demanda del usuario. El objetivo de esta librería es que pueda ser modificada por personal técnico del fabricante para mejorar los algoritmos existentes o añadir nuevos sin necesidad de modificar el software del equipo. Para esto se necesita que cada una de estas funcionalidades tengan definida una interfaz estándar. Se decide crear una clase base para cada una de estas funcionalidades: Calculador, Filtro, Detector de eventos y Entrenador. Al último se le aplica este nombre porque al proceso de obtener señales de referencia se le suele llamar entrenamiento. En la librería, los desarrolladores de los algoritmos de procesamiento deben heredar de una u otra según la funcionalidad que deseen añadir. La tecnología utilizada para el desarrollo de la aplicación, facilita la tarea de encontrar e instanciar las clases derivadas de un tipo específico. Cada una de estas clases tiene definida mediante funciones abstractas la forma de interactuar con sus derivadas. Esto garantiza que el desarrollador de estos algoritmos se vea obligado a implementarlas en las clases derivadas, garantizando esta interfaz estándar con la aplicación de visualización.

Con esta solución podemos aislar la capa de procesamientos específicos y con una única interfaz para interactuar con cada funcionalidad, la aplicación puede abstraerse de las diferencias entre unos y otros. Pero aún queda el problema de enlazar cada algoritmo con el parámetro y las variables que afecta. Esto, se decide que debe estar configurado en la propia aplicación. Anteriormente se trató sobre como la aplicación de visualización tiene que contar con la configuración de cada módulo que podría manejar y sus estructuras internas. Se aprovechará esta configuración para establecer este enlace faltante. Los parámetros fisiológicos cuentan con el nombre de su detector de eventos y/o con el nombre de su entrenador en los casos necesarios. Las variables que sean calculadas cuentan con el nombre de su calculador. Las variables que puedan ser filtradas cuentan con el nombre del filtro que se les debe aplicar.

5.1- CÁLCULO DE VARIABLES

Con lo planteado en la sección anterior, la aplicación tiene una forma de acceder a los algoritmos de cálculo de una variable, pero el proceso de utilizarlos puede tener complejidades añadidas. El cálculo de una variable usa como datos el valor de otras, las cuales pueden ser otras variables calculadas. Por lo que el orden en que se calculan se debe determinar a partir de sus dependencias. Las variables medidas por el hardware, que son la fuente original de los valores que se utilizan en los cálculos, pueden tener filtros activos si el usuario determina que está en un ambiente con alto ruido eléctrico. En estos casos, solo debe utilizarse el valor de sus mediciones luego de aplicarse el filtro correspondiente. Para satisfacer estas condiciones, se decide utilizar un árbol de cálculo, como estructura de datos que permite establecer las dependencias entre las variables calculadas y los valores de los que depende. En la figura 11 se ilustra un ejemplo de un árbol ya estructurado con el que se explicará su construcción y su empleo en el cálculo.

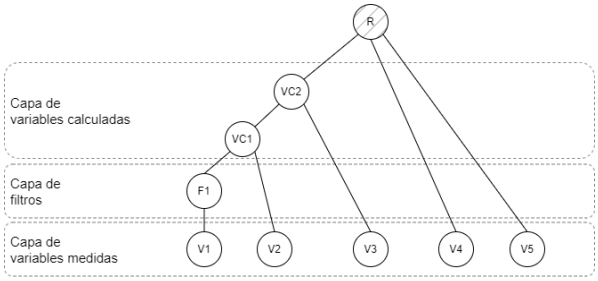

Figura 11

Árbol de ejemplo para el cálculo y filtrado de variables.

Los nodos en la figura han sido nombrados para una mejor comprensión, los que comienzan con V son las variables medidas desde el hardware, los que comienzan con F son los filtros aplicados a variables, los que comienzan con VC son las variables calculadas, y el nodo nombrado como R es el nodo raíz a partir del cual se construye el árbol. Las variables medidas son la fuente principal de valores para los cálculos por lo que siempre serán la capa final del árbol, ellas no tendrán dependencias de ninguna otra. En caso de necesitar filtrarse, tendrán en la capa inmediata a ellas su filtro correspondiente, ya que en estos casos debe utilizarse para los cálculos las variables ya filtradas. En el ejemplo la única variable que se necesitó filtrar fue V1. En la siguiente capa, se encuentran las variables calculadas, cuyos nodos hijos son los que contienen el valor necesario para su cálculo. En el caso de que una variable calculada, dependa de otra también calculada, se forma una estructura similar la de VC1 y VC2. Las variables calculadas no pueden tener dependencias circulares, o sea, si VC2 depende de VC1, VC1 no puede depender de VC2.

El algoritmo de evaluación de los valores de cada variable en el árbol comienza desde la capa inferior, adoptando los valores de las muestras recibidas por el protocolo de comunicación. Luego se aplican los algoritmos de la capa de filtrado a las variables que lo necesiten. Finalmente se aplican los algoritmos de cálculo de las variables restantes usando como datos sus dependencias del árbol. Las variables calculadas que tienen dependencia de otras variables calculadas, se calculan una vez que estas dependencias sean evaluadas. Las variables cuyos valores ya están disponibles se actualizan en sus indicadores correspondientes.

5.2- REMPLAZO DE VARIABLES

Algunas variables tienen principios similares, aunque sean medidas de forma diferente. Un ejemplo de esto es la frecuencia cardiaca, calculada a partir de la señal de electrocardiograma, la cual es medida por electrodos sobre el pecho de un paciente. La frecuencia de pulso de la presión invasiva, es medida directamente sobre las venas del paciente. Estas dos variables tienen principios de medición diferentes, pero dado que el pulso de una persona es una consecuencia de su actividad cardiaca, ambos valores deben ser similares. Esto permite que el equipo tenga un comportamiento particular sobre estos dos indicadores: si la frecuencia cardiaca deja de estar disponible (pudiera deberse a la desconexión de uno de los electrodos) se sustituye su valor en el indicador por el de la frecuencia de pulso. Cuando se recupera la medición de la frecuencia cardiaca, esta vuelve a su funcionamiento normal. A este comportamiento se le llama variable de remplazo, de forma que la frecuencia de pulso es una variable de remplazo de la frecuencia cardiaca. Para que quede claro que una variable fue remplazada se debe modificar el nombre y el color del indicador a los de la nueva variable. En la figura 12 se ilustra este comportamiento para los indicadores mencionados.

Para soportar esta funcionalidad, se decide configurar en los indicadores numéricos principales la posibilidad de tener uno o más remplazos. Los remplazos se utilizarán en el orden en que fueron configurados, por ejemplo: para que un indicador con dos remplazos utilice el segundo, tiene que estar no disponible la variable original que mostraba y el primer remplazo que se le configuró. Se define también que los indicadores que muestren más de una variable a la vez no van a soportar esta funcionalidad, debido a que definir que reemplazo se aplica a que variable y que nombre debe tomar el indicador para cada combinación posible de variables es demasiado problemático para la configuración, además de que en los parámetros típicos de estos equipos no se identifica esta necesidad.

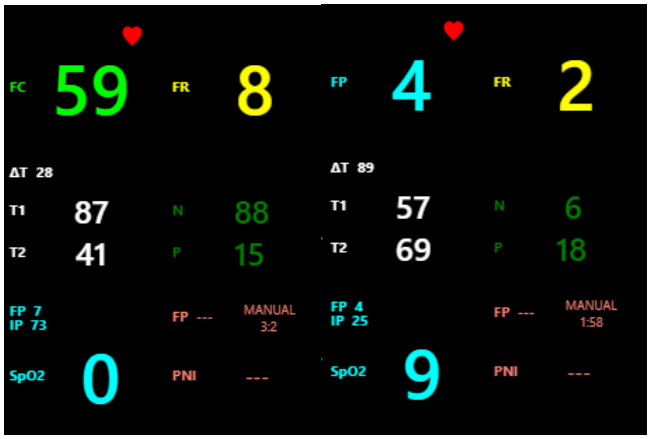

Figura 12

Remplazo de la frecuencia cardíaca por la frecuencia de pulso.

Las funcionalidades analizadas en esta sección se corresponden a la quita etapa del análisis propuesto. En la que se busca una capa de abstracción para las características que requieren un conocimiento específico del módulo. En este caso, esta capa se provee a través de la carga dinámica de una biblioteca que contiene los algoritmos particulares para cada funcionalidad. La abstracción se logra a partir de definir un estándar mediante el cual cada una de estos algoritmos interactuará con el software.

6.- APLICACIÓN DE CONFIGURACIÓN DE FÁBRICA

Correspondiendo a la última etapa del análisis, es necesario definir una estrategia de configuración para los módulos de la aplicación. La solución aplicada a los problemas de las secciones anteriores implicó que el software cuente con una configuración previamente suministrada para poder funcionar. Esto aporta una capa de desacople del software, ya que en lugar de ser una solución específica permite dar un tratamiento uniforme a los módulos del software. Sin embargo, esto también introduce algunos problemas. El hecho de tener que contar con configuración previa de un módulo para su uso, puede implicar que se necesiten actualizaciones dinámicas de las configuraciones de equipos ya distribuidos para poder usar módulos recién desarrollados. Tener que lidiar con varias versiones de equipos y, por consiguiente, varias configuraciones, puede complicar las etapas de instalación en el proceso productivo del fabricante. La variedad de estructuras y configuraciones necesarias pueden conllevar a estados incoherentes de la misma, aumentando la complejidad de la tarea de diseñar la configuración a aplicar a cada equipo.

Para mitigar estos problemas, se decide diseñar e implementar una aplicación para administrar, organizar y aplicar las distintas configuraciones para cada modelo de equipo. Esta aplicación puede ser empleada por el fabricante para el proceso de instalación de sus equipos. Además, se decide hacerlo en formato web, para potencialmente permitir que el servicio técnico pueda usarla en la reinstalación de equipos sin sacarlos del hospital donde se estén utilizando. Esta aplicación también tiene una responsabilidad importante en la validación de las configuraciones, para evitar que se apliquen de forma incorrecta. La aplicación cuenta con las siguientes organizaciones: el fabricante cuenta con varios productos, cada producto puede tener varios modelos de dispositivos, y cada modelo puede tener diferentes configuraciones. Una configuración abarca la identificación y las especificidades de cada módulo que soportará el dispositivo. Se decide utilizar este esquema ya que un fabricante puede producir distintos tipos de equipos de monitorización usando este software, cada equipo puede tener modelos diferentes de acuerdo a su hardware y las funcionalidades que presenta, cada modelo puede tener varias configuraciones de acuerdo a las particularidades de la zona la que este destinado. En la Figura 13 se muestra un ejemplo hipotético de esta distribución.

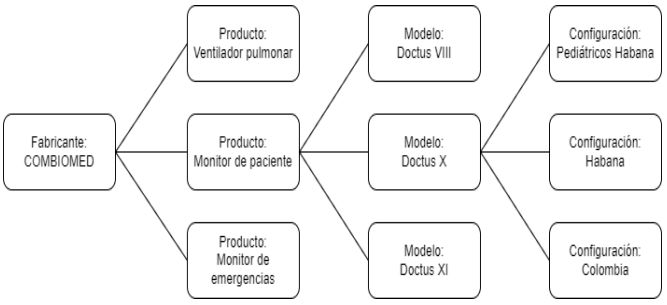

Figura 13

Ejemplo de distribución de configuraciones en aplicación de configuración.

Emplear este tipo de solución trae un riesgo de seguridad. Al tener esta aplicación de configuración de los dispositivos accesible de forma remota, podría provocar que se utilice para clonar dispositivos médicos de forma ilegal. Para evitar estas situaciones se añaden varias funcionalidades. El usuario de esta plataforma debe autenticarse para poder efectuar un proceso de configuración de un dispositivo. El software del dispositivo obtiene un código identificativo y único de su hardware con el cual se registra en la plataforma. Es necesario un certificado suministrado por el fabricante para comunicarse con la plataforma. El software del dispositivo no funcionará si no se ha configurado y activado correctamente. En la aplicación de configuración se puede visualizar cuantos equipos se han activado de cada modelo. Con estas medidas se hace muy difícil utilizar la plataforma de forma indebida. En caso de que se detecte una instalación fuera de los registros, el fabricante solo debe cambiar su certificado de comunicación para que esto no se repita.

7.- Desarrollo de la solución

La tecnología seleccionada para la implementación fue .NET 6, un ecosistema multiplataforma de Microsoft basado en el lenguaje C#. Para la aplicación visual, se utilizó la plataforma UNO, creada por la comunidad. Esta plataforma permite compilar para varios sistemas operativos, entre ellos, Windows y Linux. Se utilizó la compilación de Windows para la etapa de desarrollo de la aplicación, aumentando la productividad de los desarrolladores al ser capaces de probar la aplicación en sus computadoras. El soporte hardware utilizado para el despliegue de la aplicación fue una Raspberry Pi 4, para la cual se utilizó su sistema operativo nativo (Raspberry Pi OS), basado en Linux, por lo que esta compilación se utilizó para el despliegue en el dispositivo. Para el software se utilizó una arquitectura MVVMS (Modelo-Vista-Modelo de Vista-Servicio), ampliamente recomendada para aplicaciones de escritorio. Este patrón ayuda a separar limpiamente la lógica de una aplicación de la interfaz de usuario, aportándole una mejor mantenibilidad a la solución. Para la aplicación de configuración en fábrica se usó también la tecnología de .NET 6, pero para la interfaz visual se utilizó Angular, un marco de trabajo para el desarrollo de interfaces web, muy recomendado por su compatibilidad con la tecnología de .NET.

El proceso de desarrollo fue guiado por una metodología Agile modelo-V, en el que se implementaron pruebas unitarias y de integración para verificar requerimientos fundamentales como el protocolo de comunicación entre el software de alto nivel y los concentradores, y la persistencia en base de datos de la información de configuración e histórica. Para validar las funcionalidades desencadenadas desde la interfaz visual, se diseñaron pruebas manuales. Para verificar las funcionalidades que requerían de la interacción con los concentradores, se desarrolló una aplicación de escritorio simple, basada sobre la misma tecnología, que simula el comportamiento de estos dispositivos, de esta manera se consigue probar variados escenarios que son complicados de generar con el hardware real del dispositivo. Este proceso de pruebas y verificaciones se realizó para cada característica desarrollada del software, finalizando con la validación de los requisitos de usuario asociados. Cuando se completa una nueva característica, se realizan pruebas de integración manual para verificar que no haya afectado a las anteriormente desarrolladas. El resultado de la etapa de desarrollo se ilustra en la figura 14, en la que se muestra la pantalla de monitoreo de la aplicación ya terminada.

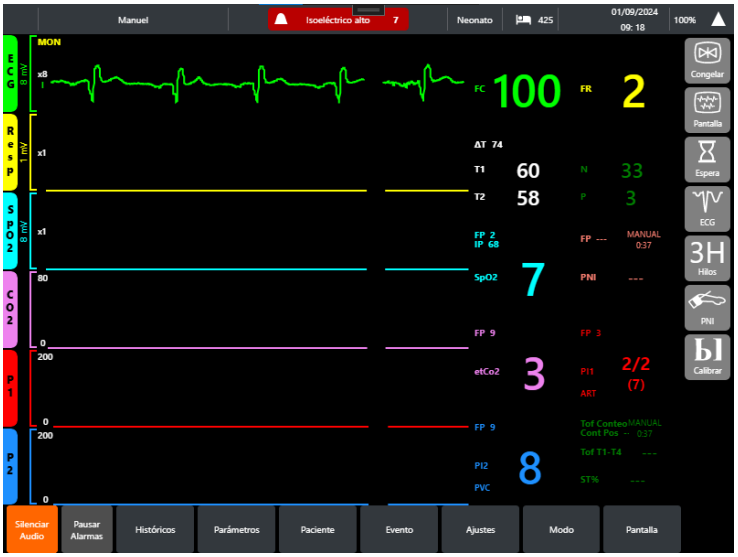

Figura 14

Pantalla de monitorización de la aplicación.

8.- Conclusiones

Se describe el análisis seguido para el diseño y la implementación de un software modular para un equipo médico, a modo de propuesta para reducir los costos de mantenimiento y desarrollo de nuevos softwares. Las dificultades de llevar un diseño monolítico a uno modular se logran solucionar con técnicas de modelado, aislamiento de los algoritmos que no se pueden

estandarizar y un sistema de configuración para abstraer al software del funcionamiento interno de sus módulos. Durante la implementación se utilizaron los enfoques recomendados en la literatura para cumplir con las regulaciones del sector.

La solución desarrollada para el equipo de monitorización de pacientes Doctus X satisface los objetivos planteados inicialmente. Se diseña un protocolo de comunicación con el propósito de soportar la identificación de distintos módulos del equipo, así como el intercambio de datos con ellos. Se diseña una estructura de configuración modular basada en la existente en los equipos previos a esta solución. Pero buscando una mayor uniformidad al integrar las funcionalidades particulares de cada módulo como variantes de configuración que pueden activarse o no. Se implementa la solución de forma que los algoritmos de cálculo y filtrado de variables y de detección de eventos fisiológicos se encuentren en una librearía fácil de modificar y sustituir para que la mejora de estos no implique realizar cambios directamente sobre el software de monitorización. Se considera esta característica desde la configuración, donde se establecen cuáles de los algoritmos deben ser cargados para aplicarse a cada variable.

Luego de conocer el volumen de información que debe ser configurado, se diseña e implementa una aplicación web para que el fabricante tenga registro y pueda modificar las configuraciones disponibles para cada equipo y sus distintos modelos. Estas son aplicadas directamente sobre los equipos en su línea de producción utilizando esta misma aplicación.

AGRADECIMIENTOS

Los autores del artículo agradecen a la empresa COMBIOMED por la oportunidad de trabajar en el proyecto y por el apoyo brindado durante el desarrollo de la solución.

REFERENCIAS

1. Liu Pu, Fels Sidney, West Nicholas, Görges. “Human Computer Interaction Design for Mobile Devices Based on a Smart Healthcare Architecture”, Advances in Computers and Software Engineering: Reviews. 2019; 2:99-13.
2. Gómez Ramírez Ana Rocío. “Evolución y Desafíos en la Verificación y Validación de Dispositivos médicos en Desarrollo”, Congreso Internacional de Investigación Academia Journals, 2024; 16(2).
3. M. McHugh, O. Cawley, F. McCaffcry, I. Richardson and X. Wang. "An agile V-model for medical device software development to overcome the challenges with plan-driven software development lifecycles," 2013 5th International Workshop on Software Engineering in Health Care (SEHC), San Francisco, CA, USA, 2013; p. 12-19
4. Khan AA, Akram MU, Butt WH, Sirshar. “M. An Enhanced Agile V-Model: Conformance to regulatory bodies and experiences from model's adoption to medical device development.” Heliyon. 2024; 10(6).
5. Kasoju, N., Remya, N.S., Sasi, R. et al. “Digital health: trends, opportunities and challenges in medical devices, pharma and bio-technology”, CSIT. 2023; 11:11–30
6. Apostolos Gerodimos, Leandros Maglaras, et al. “IoT: Communication protocols and security threats”, Internet of Things and Cyber-Physical Systems. 2023; 3:1 -13
7. Gómez Jorge, Oviedo Byron, Zhuma Emilio. “Patient Monitoring System Based on Internet of Things”, Procedia Computer Science. 2016; 83: 90-97
8. Mian Mujtaba Ali, Shyqyri Haxha, et al. “Design of Internet of Things (IoT) and Android Based Low Cost Health Monitoring Embedded System Wearable Sensor for Measuring SpO2, Heart Rate and Body Temperature Simultaneously”, Wireless Personal Communications. 2019.
9. Harvey Margaret, Seiler Amber. “Challenges in managing a remote monitoring device clinic”, ALLIED HEALTH PROFESSIONALS. 2022; 3: 4-7
10. P. Bora, P. Kanakaraja, B. Chiranjeevi et al. “Smart real time health monitoring system using Arduino and Raspberry Pi”, Materials Today: Proceedings. 2021
11. Ramy Ghanim, Anika Kaushik, Jihoon Park, Alex Abramson. “Communication protocols integrating wearables, ingestibles, and implantables for closed-loop therapies”, Device. 2023
12. Muhammad Farhan Safdar, Robert Marek Nowak, Piotr Pałka. “Pre-Processing techniques and artificial intelligence algorithms for electrocardiogram (ECG) signals analysis: A comprehensive review”, Computers in Biology and Medicine. 2024; 170.

13. Jan Maarten Schraagen , Fenne Verhoeven. “Methods for studying medical device technology and practitioner cognition: The case of user-interface issues with infusion pumps”, Journal of Biomedical Informatics. 2013; 46:181-195
14. Mindray. “Advance patient monitoring”. Sept 2024 Disponible en: https://www.mindray.com/en/products/patient-monitoring.
15 Meditech. “Multiparameter monitors”. Agosto 2024 Disponible en: https://www.meditech.com.cn/multiparameter-monitor/

CONFLICTO DE INTERESES

No existe conflicto de intereses entre los autores, ni con ninguna institución a la que cada uno está afiliado, ni con ninguna otra institución.
Las opiniones expresadas aquí son únicamente responsabilidad de los autores y no representan la posición de la Institución o las instituciones a las que están afiliados.

CONTRIBUCIONES DE LOS AUTORES

Hernán D. Pérez Villanueva: Conceptualización, Investigación, Administración de Proyecto, Software, Visualización, Redacción borrador original.

***Zuliet Medina Rodríguez*:** Conceptualización, Software, Visualización.

***Adiel Alfonso Cordoví*:** Conceptualización, Software.

***José L. Casanova Pacheco*:** Conceptualización, Software.

***Luis R. Corrales Lay*:** Conceptualización, Investigación, Recursos, Verificación.

***Alberto S. Prieto Moreno*:** Conceptualización, Investigación, Metodología, Administración de Proyecto, Recursos, Software, Visualización, Supervisión, Redacción – revisión y edición.

AUTORES

Hernán David Pérez Villanueva. Graduado de Ingeniería Automática en enero del 2022 de la Universidad Tecnológica de la Habana “José A. Echeverría” (CUJAE). Se desempeña como profesor adiestrado en el Departamento de Automática de dicha universidad y como Especialista en Informática Industrial en la empresa EMSI FARMA S.R.L. en La Habana, Cuba. Correo: hernandpv41@gmail.com, ORCID: https://orcid.org/0009-0003-5066-6556. Entre sus principales temas de interés se encuentran: desarrollo de sistemas para la industria empleando el paradigma I4.0 con tecnología .N ET, microservicios web y procesamiento de imágenes.

Zuliet Medina Rodríguez. Graduada con título de oro de Ingeniera en Automática en julio del año 2023 de la Universidad Tecnológica de La Habana “José A. Echeverría” (CUJAE). Se desempeña como docente en la facultad de Ingeniería Automática y Biomédica, además forma parte del equipo de trabajo de la empresa EMSI FARMA S.R.L. como Especialista en Informatización Industrial en La Habana, Cuba. Correo: zulietmr@gmail.com, ORCID: https://orcid.org/0000-0002-1972-5322. Entre sus principales temas de interés se encuentran: desarrollo de software (Backend) con tecnología de .Net, soluciones automatizadas, protocolos de comunicación y servicios y microservicios web.

Adiel Alfonso Cordovi, Graduado en Ingeniería en Ciencias Informáticas por la Universidad de las Ciencias Informáticas (UCI) en junio de 2018, La Habana, Cuba. Correo: adiel.alfonso.cordovi@outlook.es, ORCID: https://orcid.org/0009-0005-0332-0418 Anteriormente, trabajó como Desarrollador de Software en el Centro de Identidad y Seguridad Digital de la UCI y como Profesor Instructor de Programación en el Departamento de Técnicas de Programación de la Facultad 1 de la UCI. Además, fue Desarrollador de Software en la empresa EMSI FARMA S.R.L.

José Luis Casanova Pacheco. Estudiante de Ingeniería Automática de la Universidad Tecnológica de la Habana "José A. Echeverría" (CUJAE). Actualmente cursa el 4to año de la carrera de Ingeniera Automática. Correo: casanovapachecojl@otulook.es, ORCID: https://orcid.org/0009-0009-1241-6031. Entre sus principales temas de interés se encuentran: desarrollo de sistemas para la industria empleando el paradigma I4.0 con tecnología .NET y microservicios web.

Luis Raydel Corrales Lay. Graduado de Ingeniería Biomédica en julio del año 2016 del Universidad Tecnológica de la Habana "José A. Echeverría" (CUJAE). Se desempeña como Director de Producciones Electrónicas en COMBIOMED Tecnología Médica Digital en La Habana, Cuba. Correo: raydelcl91@gmail.com, ORCID: https://orcid.org/0009-0002-2319-4015. Entre sus principales temas de interés destacan las tecnologías electrónicas, microelectrónicas y softwares aplicadas a las ciencias biomédicas.

Alberto Sergio Prieto Moreno. Graduado con título de oro de Ingeniería en Automática en julio del año 2004 de la Universidad Tecnológica de La Habana "José A. Echeverría" (CUJAE). Doctor en Ciencias Técnicas, de la propia universidad en el año 2013. Se desempeña como profesor Titular a tiempo parcial y subdirector de la empresa EMSI FARMA S.R.L. en La Habana, Cuba. Correo: albprietom@gmail.com, ORCID: https://orcid.org/0000-0001-9907-885X. Entre sus principales temas de interés se encuentran: análisis de datos, diagnóstico de fallos, desarrollo de sistemas para la industria empleando el paradigma I4.0 con tecnología .NET y microservicios web.