

UNIVERSIDADE DE SANTIAGO DE COMPOSTELA
ESCOLA TÉCNICA SUPERIOR DE ENXEÑARÍA



**Interface para a visualización e xestión do seguimento
domiciliario de pacientes.**

MEMORIA
TRABALLO FIN DE GRAO

Presentada por:
José Ángel Piñeiro Souto

Dirixida por:
Dr. Paulo Félix Lamas

Codirixida por:
Tomás Teijeiro Campo
Ana Gándara Núñez

Santiago de Compostela, Xullo 2011

Índice xeral

1. Introdución	1
1.1. Plataforma Servando	2
1.2. O problema: Xestión da información	3
1.3. Obxectivos do proxecto	3
1.4. Organización do documento	4
2. Xestión do proxecto	5
2.1. Xestión de riscos	5
2.1.1. Riscos de organización e xestión	6
2.1.2. Riscos de desenvolvemento	6
2.1.3. Riscos de persoal	7
2.2. Metodoloxía de desenvolvemento	7
2.2.1. Ferramentas de xestión	10
2.2.2. Aplicación da metodoloxía	13
2.3. Planificación temporal	13
2.3.1. Fase de inicio	14
2.3.2. Fase de elaboración	14
2.3.3. Fase de construcción	14
2.3.4. Fase de transición	15
2.3.5. Fase de conclusión	15
2.4. Xestión da configuración	15
2.4.1. Control do código fonte	16
2.4.2. Control da documentación	16

2.5. Análise de custos	18
3. Análise de requisitos	19
3.1. Definicións	19
3.2. Casos de uso	21
3.2.1. Descripción de actores	21
3.2.2. Descripción de casos de uso	21
3.3. Restricións de deseño	24
3.4. Requisitos funcionais	25
3.5. Requisitos de proxecto	33
3.6. Requisitos de calidade	33
3.7. Requisitos de almacenamento	34
3.8. Organización de <i>sprints</i>	34
4. Arquitectura, tecnoloxías e ferramentas	39
4.1. Alternativas dispoñibles	39
4.2. Arquitectura do sistema	41
4.3. Ferramentas de deseño	45
4.4. Ferramentas de desenvolvemento	49
4.4.1. Java SE	50
4.4.2. Hibernate	50
4.4.3. PostgreSQL	53
4.4.4. Javascript	54
4.4.5. JSON	54
4.4.6. Ajax	54
4.4.7. Librarías externas	55
4.5. Ferramentas de validación	56
4.5.1. JUnit	56
4.5.2. Avaliación heurística	56
5. Deseño e implementación	59
5.1. Modelo de datos	59

5.2.	Arquitectura da información	61
5.3.	Deseño da interface de usuario	63
5.3.1.	Estrutura da interface	63
5.3.2.	Compoñentes da interface	63
5.4.	Deseño e implementación do servidor	73
5.4.1.	Motor de persistencia	73
5.4.2.	Controladores e Adaptador JSON	76
5.5.	Deseño e implementación do cliente	77
5.5.1.	Núcleo do cliente	77
5.5.2.	Subsistema de monitorización	80
5.5.3.	Subsistema de xestión de pacientes	83
5.5.4.	Módulos auxiliares	85
6.	Probas e validación	87
6.1.	Preparación das probas	87
6.2.	Deseño das probas	88
6.3.	Avaliación heurística	102
6.4.	Validación	104
7.	Conclusóns	113
A.	Manual de usuario	117
A.1.	Instalación	117
A.2.	Axuda	118
A.2.1.	Xestión de usuarios	118
A.2.2.	Xestión de pacientes	119
A.2.3.	Visualización do seguimento	120
A.2.4.	Visualización de eventos e episodios	120
A.2.5.	Visualización de parámetros	122
A.2.6.	Visualización de electrocardiograma	123

Capítulo 1

Introducción

A evolución que están a experimentar as tecnoloxías da información e as comunicacións (TIC) nos últimos anos, xunto co seu grao de penetración nos fogares, que en Galicia xa ronda o 55 % [9], fai que o seu uso e aproveitamento na vida cotiá sexa cada vez maior. A saúde non é en absoluto aldea a este feito, e constantemente aparecen novas propostas que tratan de mellorar os procesos clásicos de tratamento e coidado clínico mediante o uso da tecnoloxía. En particular, a telemedicina, consistente na provisión de servizos médicos de forma remota, intenta fazer fronte a un conxunto de problemas comúns ós actuais sistemas públicos de saúde, dos cales podemos destacar: i) o progresivo envellecemento da poboación, co conseguinte aumento da prevalencia de enfermidades crónicas; ii) unha organización da súa actividade baseada no tratamiento puntual e episódico de enfermidades en fase aguda; iii) as dificultades dunha axeitada asimilación e incoporación do novo coñecemento á rutina asistencial. A todos estes problemas podémoslle sumar neste momento o contexto de grave crise económica, que obriga a unha redución de gastos en tódolos ámbitos, incluído o sanitario, o que desgraciadamente se soe traducir nun empeoramento na calidade do servicio ofrecido. A telemedicina tamén podería aportar vantaxes neste aspecto, en tanto que unha vixilancia e coidados más personalizados e continuos reducirían o número de episodios de gravidade sufridos polos pacientes, e que son os que inducen un maior custo no seu tratamento.

Na actualidade existe xa un gran número de propostas de sistemas que traballan neste sentido, principalmente para o seguimento de anciáns, de persoas diminuídas que viven soas, de enfermos crónicos, ou de pacientes que sufrieron procesos que requieren un seguimento posterior (por exemplo un Infarto Agudo de Miocardio). Neste sentido, o noso punto de partida será a plataforma de seguimento domiciliario Servando [31], que define un novo marco para o desenvolvemento de sistemas de telemedicina.

Este proxecto enmárcase no conxunto de actividades que integra o Proxecto de Investigación “AI-SENIOR: Razonamiento temporal y minería de datos en sistemas de monitorización ubicua para el cuidado de las enfermedades de EPOC y EC” (Ref. TIN2009-14372-C03-03).

1.1. Plataforma Servando

Servando é unha plataforma distribuída e extensible que pretende resolver unha serie de problemas comúns a tódolos sistemas de telemedicina. Entre as súas principais características, podemos citar as seguintes:

1. Permite a **planificación temporal** dun conxunto de actuacións médicas programadas a modo de axenda de seguimento para un paciente, mediante unha linguaxe propia de descripción de protocolos de seguimento médicos.
2. **Encapsula a funcionalidade** relativa á realización das actuacións médicas (monitorizacions, realización de cuestionarios, administracions terapéuticas, etc.) nun conxunto de servicios médicos, que poden ser incluídos ou eliminados da plataforma dinamicamente.
3. Ofrece un conxunto de **interfaces de programación** (APIs) para o desenvolvemento de novos servizos médicos de xeito totalmente independente á plataforma, na que poderán integrarse con posterioridade de forma transparente.
4. Resolve o problema das comunicacions entre o domicilio do paciente e o centro médico, mediante un módulo flexible e extensible de intercambio de mensaxes bidireccionais, que permite a transmisión sinxela de calquera tipo de información.

Na figura 1.1 pode verse a arquitectura de alto nivel da plataforma, resaltando o ámbito de actuación do presente Traballo de Fin de Grao. Hai que destacar que a plataforma está actualmente sendo empregada para o seguimento de pacientes con insuficiencia cardiaca e pacientes con enfermidade cardiovascular.

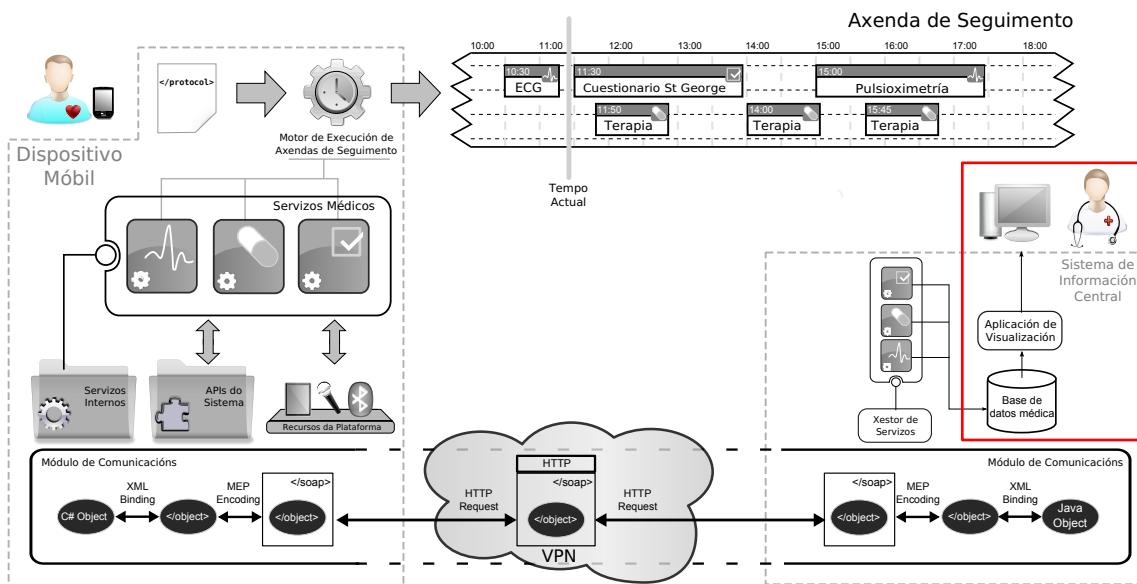


Figura 1.1: Arquitectura de Servando.

1.2. O problema: Xestión da información

Servando aborda moitos dos problemas típicos que aparecen de xeito recurrente á hora de desenvolver sistemas de telemedicina. Porén, dá lugar a un grave problema de xestión da información: Os diferentes servizos médicos incluídos na plataforma enviarán e recibirán mensaxes de moi diversa natureza e empregando distintos tipos de datos (dende fragmentos de sinais fisiolóxicos ata imaxes, pasando por resultados de cuestionarios ou probas médicas puntuais), que deberán ser gardados convenientemente no sistema de información que permitirá posteriormente a súa axeitada visualización por parte do persoal clínico. Por outra banda, a análise de toda esta información supón unha sobrecarga de traballo para o persoal médico, engadindo os pacientes baixo seguimento domiciliario ós pacientes que debe atender en consulta. Servando aborda este problema do lado do paciente, incorporando unha etapa de procesamento dos datos adquiridos por parte dos servizos médicos, que pretende construir información de máis alto nivel e transmitir únicamente aquela información relevante no seguimento do paciente. Así todo, o persoal médico precisa mecanismos de visualización que permitan unha rápida identificación e caracterización do ocorrido durante longos periodos de tempo.

O presente Traballo de Fin de Grao pretende fazer fronte a estes problemas de xestión da información. Por unha parte, establecendo unha estratexia e mecanismos para o almacenamento eficiente, e sobre todo extensible, de toda a información que se recibirá como resultado do seguimento remoto, axustándose ó modelo de transmisión da información establecido por Servando. Por outra banda, permitindo a recuperación e visualización desa información de forma eficiente e visual, centrándose nos eventos relevantes pero permitindo un acceso sinxelo e intuitivo ó resto da información que puidese requirir o persoal clínico.

1.3. Obxectivos do proxecto

Os obxectivos do presente proxecto son os seguintes:

- Realizar a análise, deseño e implementación dunha ferramenta de persistencia para a plataforma Servando, que:
 - Permite o almacenamento dinámico de distintos tipos de datos provintes de diferentes fontes, e a priori desconecidos.
 - Facilite a adición de novos tipos de información, minimizando os cambios necesarios para incorporalos.
 - Teña boas prestacións en canto a rendemento e soporte de concurrencia.
 - Sexa compatible con tódolos módulos da plataforma actualmente implementados.
- Realizar a análise, deseño, e implementación dunha interface de visualización para o seguimento domiciliario de pacientes, que:
 - Permite realizar de forma gráfica e sinxela tódalas tarefas involucradas no seguimento domiciliario de pacientes, en particular pacientes con enfermidade cardiovascular e

pacientes con insuficiencia cardiaca.

- Empregue metáforas visuais para a presentación de información temporal que faciliten a análise dos datos a diferentes granularidades temporais, dende segundos a semanas, pasando por minutos, horas e días.
 - Basando o seu funcionamento en amosar e permitir contextualizar os principais eventos de interese detectados durante o seguimento, permita de forma sinxela acceder a toda a información relacionada que puidera ser de interese nun contexto determinado.
 - Sexa intuitiva, ergonómica, e visualmente atractiva para o usuario.
- Realizar unha exploración e experimentación de tecnoloxías profunda, comprobando as capacidades ofrecidas polas últimas ferramentas de desenvolvemento.
 - Despregar a ferramenta nun entorno real de seguimento de pacientes, e poñela a disposición do persoal clínico para o seu uso en produción.
 - Finalizar tódalas etapas de desenvolvemento antes do 11 de Xullo de 2011.

1.4. Organización do documento

Ó longo deste documento expónense os procedementos e metodoloxías empregados para o cumprimento de cada un dos obxectivos deste proxecto.

O capítulo 2 aborda os aspectos realltivos á xestión do proxecto, os riscos que presenta, a metodoloxía de desenvolvemento empregada, a planificación temporal e a xestión da configuración.

No capítulo 3 preséntase o proceso de análise de requisitos, partindo do conxunto de casos de uso do sistema para obter unha lista de requisitos iniciáis do proxecto e as súas restriccións.

O capítulo 4 plantéxase a arquitectura global do sistema. Comézase realizando unha análise das diferentes posibilidade á hora de levalo a cabo, e a continuación preséntase a arquitectura do sistema e o conxunto de ferramentas utilizadas para o deseño, desenvolvemento e validación do sistema.

No capítulo 5 afóndase nos aspectos de deseño e implementación da ferramenta, partindo da arquitectura proposta no capítulo anterior, e detallando os diferentes artefactos que a compoñen.

No capítulo 6 preséntanse as probas realizadas para cada requisito, así como a metodoloxía seguida para comprobar a validez de cada un deles, e o seu grao de cumprimento.

O capítulo 7 expónense as conclusíons do desenvolvemento, resumindo o grao de cumprimento dos diferentes obxectivos.

Capítulo 2

Xestión do proxecto

Durante a execución de calquera proxecto é necesario aplicar unha serie de técnicas ou metodoloxías que, ademais de axudarnos a garantir a calidade do produto final, nos axuden a levar a cabo todo o traballo necesario para cumplir co alcance dentro dos límites de tempo e custo establecidos. Entre os aspectos más importantes a ter en conta están a xestión de riscos, a selección dunha metodoloxía de desenvolvemento que se axeite a eses riscos e ós obxectivos expostos, a planificación temporal dos diferentes procesos involucrados na metodoloxía, a realización de estimacións precisas de custo e prazos, ou a xestión da configuración de tódolos produtos obtidos en calquera das etapas do desenvolvemento. Todos estes puntos serán obxecto de discusión neste capítulo, que tratará de explicar de xeito preciso as decisións tomadas referentes á xestión do proxecto.

2.1. Xestión de riscos

Cando falamos dun proxecto de desenvolvemento de software, podemos definir un risco como un evento ou condición incerta, que, en caso de ocorrer pode ter consecuencias desfavorables nos obxectivos do mesmo. Unha das tarefas más importantes ao inicio dun proxecto é a de identificar e eliminar ou minimizar as posibles fontes de risco, para evitar que poidan ameazar a finalización satisfactoria do proxecto. Hai que ter en conta que este é un proxecto académico enmarcado nun programa de investigación, cunha alta compoñente de experimentación, no que un dos obxectivos é poñer a proba a tecnoloxía e comprobar en que medida esta pode facilitarnos o traballo e satisfacer as nosas necesidades, o que implicitamente introduce unha compoñente de risco clara debido á falta de experiencia e o desconecemento das ferramentas, tanto a nivel de uso como de capacidade e rendemento das mesmas.

Para analizar os riscos seguirase a metodoloxía exposta por Fairley [13], que se basea na cuantificación dos diferentes riscos detectados tendo en conta a probabilidade de que sucedan e a criticidade do seu suceso para os obxectivos do proxecto. Unha vez se teña unha lista de riscos concretos, disporanse unha serie de medidas de prevención que traten de minimizar tanto a probabilidade de que ocorran como o seu posible impacto.

A continuación móstranse os riscos detectados que se cre preciso considerar para asegurar o correcto desenvolvemento do proxecto, clasificados en categorías, e priorizados en base ao seu posible impacto.

2.1.1. Riscos de organización e xestión

- Cambios no alcance durante o transcurso do proxecto
 - *Descripción:* Dende o inicio do proxecto prevese que os requisitos van sufrir cambios, dada a alta compoñente de experimentación existente.
 - *Probabilidade:* Alta
 - *Criticidade:* Moi alta
 - *Medidas de prevención tomadas:* Para minimizar o impacto de posibles variacións nos requisitos iniciais do proxecto, deberase ter este factor especialmente en conta á hora de seleccionar unha metodoloxía de desenvolvemento.
- Incumprimento de prazos de entrega
 - *Descripción:* Dada a pouca experiencia en tarefas de xestión que se acumula ata o momento en proxectos de envergadura similar, é difícil realizar estimacións temporais realistas, polo que existe o risco de non cumplir os prazos establecidos para as diferentes entregas que se vaian realizando.
 - *Probabilidade:* Media
 - *Criticidade:* Moi alta
 - *Medidas de prevención tomadas:* Para evitar problemas deste xénero, involucrarase altamente ó cliente, mantendo reunións en intervalos de tempo curtos, que non permitan grandes atrasos. Farase tamén uso ferramentas que axuden a estimar duracións, e tratarase de non errar por defecto nas estimacións realizadas ó comezo do proxecto, para ir afinando progresivamente a precisión das mesmas.

2.1.2. Riscos de desenvolvemento

- Inexistencia de librarías que ofrezan a funcionalidade requirida
 - *Descripción:* Durante a construcción do proxecto será preciso contar cunha serie de librarías que nos sirvan de apoio á hora de desenvolver. É previsible que as solucións existentes no mercado non ofrezan toda a funcionalidade requirida, o que pode dar lugar a diversos problemas como a imposibilidade de implementar unha certa característica coa linguaxe ou bibliotecas utilizadas e obrigarnos a buscar outras ou incluso a implementalas para conseguir a funcionalidade desexada.
 - *Probabilidade:* Media
 - *Criticidade:* Moi alta

- *Medidas de prevención tomadas:* Para evitar problemas deste tipo deberase realizar unha análise previa das diferentes alternativas existentes, valorando especialmente as posibilidades de extensión ofrecidas.
- Viabilidade técnica da solución proposta
 - *Descripción:* Dado que non temos datos acerca do rendemento de sistemas similares, pode ocorrer que este non satisfaga as expectativas.
 - *Probabilidade:* Media
 - *Criticidade:* Alta
 - *Medidas de prevención tomadas:* Deberase realizar unha pequena análise de viabilidade que asegure que as tecnoloxías que se van a utilizar ofrecen un rendemento axeitado. Ademais disto, farase unha análise simultánea durante a execución do proxecto que nos permita levar a cabo acción correctoras en caso de ser necesarias.

2.1.3. Riscos de persoal

- Falta de experiencia
 - *Descripción:* Tanto a linguaxe como a maioría das tecnoloxías a empregar son novas para nós. Esta falta de experiencia pode afectar negativamente tanto á exactitude das estimación realizadas como á calidade do produto final.
 - *Probabilidade:* Media
 - *Criticidade:* Alta
 - *Medidas de prevención tomadas:* Durante o desenvolvemento, tratarase de implemetar en primeiro lugar aquelas funcionalidades ou partes do sistema que acarrenen unha menor dificultade para, deste xeito, familiarizarnos coas novas ferramentas a medida que o proxecto transcorre e sen ter que inverter unha gran cantidade de tempo ó inicio do proxecto exclusivamente a formación.

2.2. Metodoloxía de desenvolvemento

Un dos primeiros pasos que debemos dar á hora de embarcarnos nun proxecto, é o de elixir unha metodoloxía de desenvolvemento adecuada, que estableza un camiño para levalo a cabo de maneira sistemática e proporcione uns estándares de traballo que aseguren a finalización exitosa do mesmo. Porén, esta non é unha tarefa trivial, e plantexa unha serie de dificultades de sobre coñecidas no mundo da enxeñería informática.

Unha delas é a de escoller entre unha metodoloxía predictiva fundamentada no establecemento e posterior seguimento dun plan de proxecto estrito e pouco flexible, ou unha metodoloxía áxil baseada na adaptatividade. As metodoloxías predictivas baséanse, como o seu propio nome indica, na existencia dun plan de traballo predecible, e polo tanto, en contar cun conxunto estable de requisitos, o que fai que sexan pouco tolerantes a cambios. As metodoloxías áxiles, pola contra,

permiten un mellor manexo de riscos e responden mellor en situacíons onde os requisitos están pouco definidos, pois dende un principio se asume que se van a producir cambios e establecese un marco de traballo que se adapta a tales circunstancias.

Este proxecto vai ser realizado de xeito individual por alguén con escasa experiencia na xestión de proxectos de tamaño considerable, como é o caso, e como xa dixemos, ten unha compoñente de innovación e experimentación importante. Isto fai que non resulte doado caracterizar o sistema e sexa difícil proporcionar ou probar exemplos reais de sistemas similares, o que resulta nun aumento da probabilidade de que se produzan cambios durante o transcurso do proxecto. Analizando tales circunstancias parece razoable pensar que debemos decantarnos por unha metodoloxía de carácter adaptativo, que garanta a resposta ó cambio.

Dentro das diferentes metodoloxías áxiles existentes, para a xestión deste proxecto optouse polo uso de *Scrum* [26], un marco de traballo para a xestión de software que propón unha serie de prácticas a seguir e un conxunto de ferramentas a utilizar, a fin de proporcionar unha serie de beneficios como a xestión regular das expectativas do cliente, flexibilidade e adaptación a cambios respecto das súas necesidades ou dos requisitos, e unha mitigación sistemática dos riscos do proxecto. En *Scrum* realizanse entregas parciais e regulares do producto final, priorizadas polo beneficio que aporta ao receptor do proxecto. Por isto, está especialmente indicado para proxectos coma este, nos que se necesita obter resultados pronto, onde os requisitos son cambiantes ou están pouco definidos e onde a innovación, a flexibilidade, e a produtividade son fundamentais.

Pode parecer, á vista dos riscos do proxecto e do desenvolvemento iterativo proposto, que outras metodoloxías clásicas como a espiral [30] poderían ser aplicadas obtendo resultados satisfactorios. Porén, aplicando un ciclo de vida en espiral non sempre se obteñen partes funcionais do producto en cada iteración, o que neste proxecto se considera fundamental, non se involucra tan constantemente ó cliente, e realiza unha xestión máis enfocada a evitar os riscos que a proporcionar valor ó cliente.

A continuación exporanxe con un pouco máis de detalle os fundamentos de *Scrum*, así como as actividades e as ferramentas propostas por esta metodoloxía, para seguidamente presentar as ferramentas de xestión que nos permitirán aplicala de xeito rigoroso no marco deste proxecto.

Fundamentos

O funcionamento de *Scrum* está baseado nos seguintes puntos:

- Desenvolvemento iterativo e incremental: O proxecto planifícase en pequenos bloques temporais ou iteracións, comunmente chamados *sprints*. En cada *sprint* repítense un proceso de traballo similar para proporcionar un resultado completo sobre o producto final, engadindo novos requisitos e cumprindo novos obxectivos.
- Priorización dos requisitos: É necesario priorizar os requisitos de maneira regular, en función do seu valor para o cliente e o custo de desenvolvemento nese momento.
- Control empírico do proxecto: Asúmese que existe un horizonte de predición das variables do proxecto dado, e que vai haber cambios. O proceso de control baséase na inspección e

na adaptación regular en función dos resultados que se van obtendo do propio contexto do proxecto.

- Potenciación do equipo: O equipo de traballo é autoxestionado, e ten a responsabilidade de decidir que tarefas se realizan en cada iteración, e a autoridade para xestionar de que xeito facelo.
- Colaboración co cliente: O cliente está altamente implicado no proceso de desenvolvemento, colaborando co equipo para planificar, revisar e detallar os obxectivos de cada iteración.
- Prazos fixos (*timeboxing*): Fíxase un tempo máximo para conseguir uns obxectivos ou realizar unhas tarefas, non permitíndose a modificación do prazo establecido inicialmente. Isto favorece a priorización dos obxectivos e tarefas, e forza a toma de decisiones.

Para lograr todo isto, *scrum* propón as seguintes actividades e ferramentas:

Actividades

- Planificación da iteración (*Sprint Planning*): O primeiro día de cada *sprint*, seleccionanse os requisitos a realizar nel. A continuación planifícase a iteración e defínese a lista de tarefas a realizar para completala, creando a *lista de tarefas da iteración*.
- Execución da iteración (*Sprint*): Durante o *sprint* realízanse sincronizacíons periódicas entre os membros do equipo e actualízase o estado da lista de tarefas da iteración.
- Demostración dos requisitos (*Sprint Review*): Ó final do *sprint* realiza unha reunión na que se mostran ó cliente os requisitos completados. En función dos resultados mostrados e dos cambios que pudiera haber no contexto do proxecto ou nos obxectivos do mesmo, realízanse as adaptacións necesarias.
- Retrospectiva (*Sprint Retrospective*): Co obxectivo de mellorar a produtivididade, o equipo analiza que cousas hai que mellorar, que cousas funcionan ben, e que problemas poden impedir o adecuado progreso do proxecto.
- Replanificación: Nas reunións de planificación trabállase conxuntamente co cliente, engadindo, modificando e cambiando a prioridade dos requisitos en función das súas necesidades.

Ferramentas

- A **lista de requisitos priorizada**, que representa as expectativas do cliente respecto ós obxectivos, e que contén unha serie de requisitos de alto nivel, que normalmente se poden expresar como *casos de uso*.
- Para cada *sprint*, unha **lista de tarefas da iteración**, coas tarefas que o equipo considera necesarias para completar os requisitos propostos, e que se compromete a demostrar ó cliente ó remate da iteración.

- **Gráficos de traballo pendente** que permitan ver a velocidade á que se traballa e extrapolar se se poderá completar o traballo no tempo estimado.

2.2.1. Ferramentas de xestión

Unha vez seleccionada unha metodoloxía, existen numerosas ferramentas que poden ser de utilidade á hora de realizar a xestión do proxecto. Unha das más amplamente utilizadas son os Diagramas de Gantt, que permiten representar e xestionar de xeito eficiente as diferentes fases, tarefas e actividades programadas no marco dun proxecto. A maioria de ferramentas existentes a día de hoxe permiten, ademais, a súa integración con outras tarefas como a xestión de recursos ou a xestión de custos. Porén, o presente proxecto vai ser levado a cabo por unha única persoa, seguindo unha metodoloxía áxil, e ánda que podemos usar Diagramas de Gantt para elaborar un plan de traballo de alto nivel, estes non se adaptan do mellor xeito á metodoloxía elixida.

Como se expuxo no apartado anterior, *scrum* propón unha serie de actividades e un conxunto de ferramentas que idealmente deberían ser realizadas de xeito sistemático por medio dalgúnha ferramenta de xestión que permita a planificación do proxecto en base a *sprints* a medida que este avanza, e nos ofrece unha maneira de automatizar a xestión das listas de requisitos de cada iteración e a lista de requisitos global do proxecto.

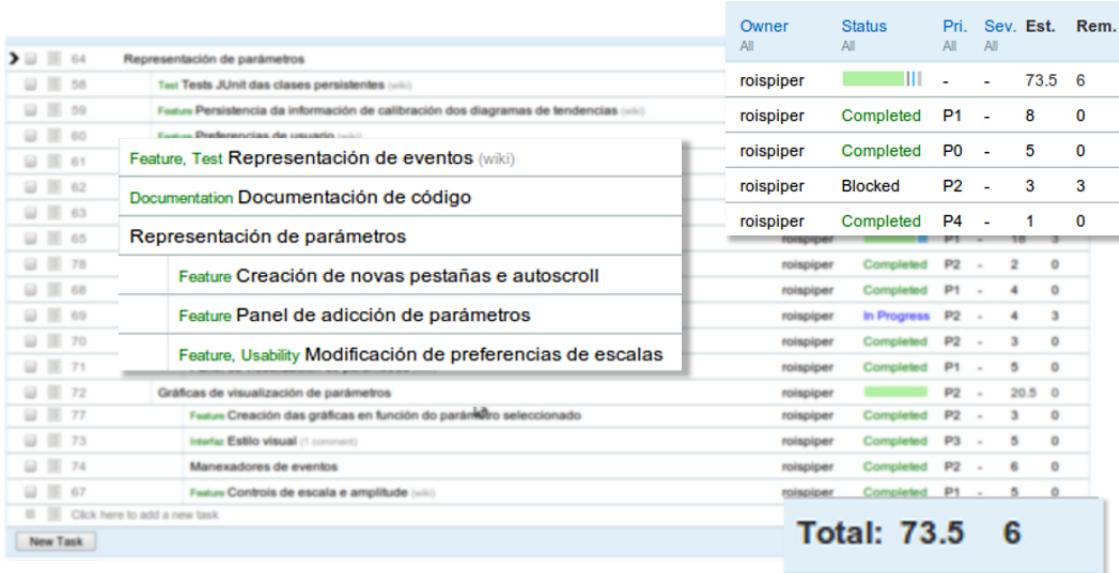
Por suposto, existen ferramentas deseñadas especialmente para esta tarefa. A que nós utilizaremos durante o transcurso do proxecto, e que se explicará en detalle a continuación, será *Acunote*.

Acunote

Acunote é unha ferramenta web de uso gratuito orientada á xestión de proxectos baixo a metodoloxía *Scrum*, e conta cunha serie de funcionalidades que cubren os diferentes aspectos a ter en conta durante a execución dun proxecto coma este, das que se falará a continuación.

- Unha das funcionalidades más importantes é a de **crear a lista de requisitos priorizada** do producto (*product backlog*), que nos permite xestionar de xeito eficiente o conxunto de requisitos do proxecto. Esta lista defíñese o comezo do proxecto, e non ten por que estar completa nese momento, senón que se vai adaptando durante o desenvolvemento, engadíndolle novos requisitos e modificando os existentes.
- Unha vez feito isto, pódese proceder á **creación de sprints**. Isto é tan simple coma establecer a súa duración (de 1 a 4 semanas), e seleccionar da lista de requisitos global as tarefas que durante el se realizarán. Débese tamén establecer, para cada tarefa do *sprint*, as horas de traballo restantes ata a súa finalización. A figura 2.1 amosa un dos *sprints* realizados durante a execución do proxecto.

Como se pode ver na parte esquerda da figura, cada tarefa ten un identificador, un nome, e pode ser etiquetada en diferentes categorías (interface, documentación, funcionalidade, etc.), o que permite ver o progreso en cada unha delas, e conta cun espazo propio no que se permite a introdución de calquera tipo de documentación de máis baixo nivel que o usuario



The screenshot shows a software interface for managing a sprint backlog. On the left, a tree view lists tasks under various categories like 'Representación de parámetros', 'Feature', 'Test', and 'Documentation'. A specific task, 'Feature, Test Representación de eventos', is highlighted. On the right, a table provides detailed information for each task, including owner ('roispiper'), status (e.g., 'Completed', 'In Progress'), priority ('P1' to 'P4'), severity ('-'), estimated time ('Est.'), and remaining time ('Rem.'). Below the table, a summary box displays 'Total: 73.5 6'. At the bottom left, there's a 'New Task' button.

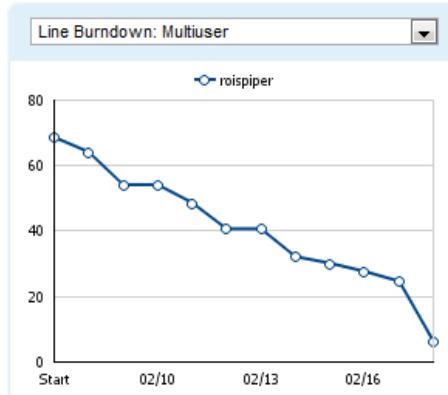
Owner	Status	Pri.	Sev.	Est.	Rem.
All	All	All	All	All	
roispiper	In Progress	P1	-	73.5	6
roispiper	Completed	P1	-	8	0
roispiper	Completed	P0	-	5	0
roispiper	Blocked	P2	-	3	3
roispiper	Completed	P4	-	1	0
roispiper	In Progress	P1	-	18	0
roispiper	Completed	P2	-	2	0
roispiper	Completed	P1	-	4	0
roispiper	In Progress	P2	-	4	3
roispiper	Completed	P2	-	3	0
roispiper	Completed	P1	-	5	0
roispiper	In Progress	P2	-	20.5	0
roispiper	Completed	P2	-	3	0
roispiper	Completed	P3	-	5	0
roispiper	Completed	P2	-	6	0
roispiper	Completed	P1	-	5	0

Total: 73.5 6

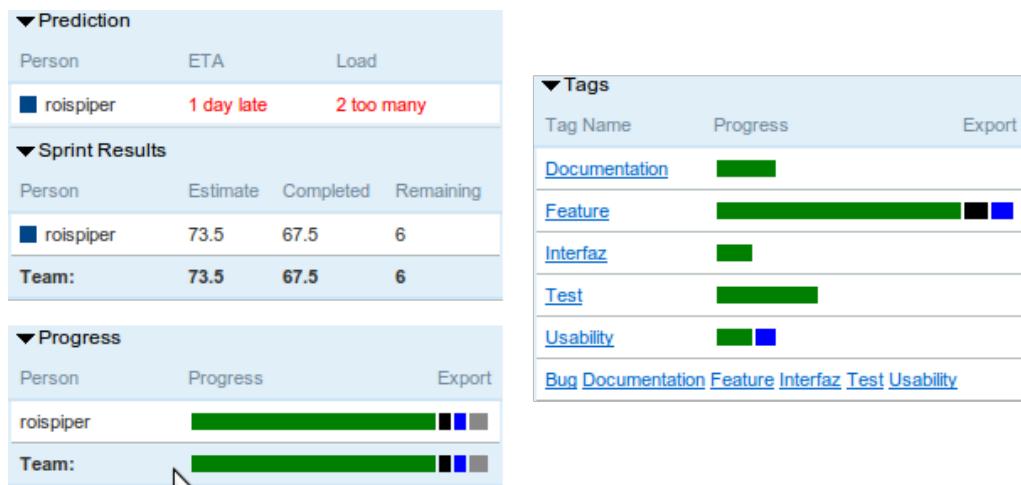
Figura 2.1: Lista de tarefas dun sprint (*sprint backlog*) en Acunote

poida precisar. Esta característica será especialmente útil no noso caso para a compilación da documentación técnica e a realización da memoria. Na parte dereita móstrase o estado das tarefas, os seus responsables, a súa prioridade, e a cantidade de traballo que é necesario realizar para completala, xunto co traballo que se estimou nun principio. Na parte inferior móstrase o total de traballo restante para rematar tódalas tarefas do *sprint*.

figura 2.1

Figura 2.2: Gráfico de traballo pendente dun dos *sprints* xestionados mediante Acunote

- Durante a **execución dun *sprint***, faise unha actualización diaria do traballo realizado e do traballo restante para cada tarefa da lista. Isto permite, se é preciso, a reestimación da duración das tarefas a medida que estas se van completando, así como un maior control do traballo pendente e a capacidade de prever se ese traballo se poderá completar no tempo

Figura 2.3: Estadísticas dun *sprint* en Acunote

Recorded Time Stats						
	02/07	02/08 – 02/13	02/14	02/15	02/16	02/17
Est.	5					
Rem.			3	2		1
Own.	ÁPS					0
Stat.	NS		IP			C

Figura 2.4: Detalle dunha das tarefas realizadas e xestionadas mediante Acunote

restante. Acunote ofrece toda esta información de xeito visual a través de gráficos de trabalho pendente (figura 2.2) que se actualizan a medida que se van producindo cambios nas duracións das tarefas.

Por outra banda, como se amosa na figura 2.3 móstranse unha serie de predicións sobre o cumprimento dos prazos establecidos e da cantidade de trabalho que se poderá completar ata a finalización do *sprint* en base ao ritmo de trabalho actual.

Na figura 2.4 móstrase a evolución dunha das tarefas realizadas durante o proxecto. Pódese ver como no momento da súa creación se lle engaden 5 horas de trabalho pendente e se lle asigna o estado *NS* (*not started* ou non iniciada). Días despois comézase coa tarefa co que o seu estado pasa a ser *IP* (*in progress* ou en progreso) e ese mesmo día complétanse 2 horas de trabalho quedando 3 restantes. En días sucesivos segue a traballarse na tarefa ata que finalmente é completada e o seu estado se establece como *C* (*completed* ou completada).

- Unha vez rematado un *sprint*, ofréncense unha serie de resultados en forma de estadísticas que poden ser de axuda para a planificación de *sprints* futuros. Un dos más útiles é a carga de trabalho, pois permite comprobar a exactitude das estimacións realizadas, en función do trabalho previsto para cada tarefa ó comezo do *sprint* e dos cambios realizados nas diferentes tarefas durante a súa realización.

Se quedaron tarefas sen completar, estas pódense mover á lista de requisitos xerais ou directamente a un novo *sprint*.

2.2.2. Aplicación da metodoloxía

Como xa comentou anteriormente, a elección e implantación dunha metodoloxía de traballo non é unha tarefa trivial. Incluso cando xa se optou polo establecemento dun marco de traballo concreto, é necesario axustalo e adaptalo ás características propias da organización e do proxecto. A continuación expoñeranse unha serie de decisións tomadas neste aspecto, así como as razóns que levaron a adoptalas, para garantir a aplicación correcta e eficaz de *scrum* a este proxecto.

- Dado o alto grao de participación do cliente e o firme compromiso mostrado tanto por este como polo equipo de traballo decidiuse que a duración dos diferentes *sprints* do proxecto fose de unha semana para aqueles *sprints* relativos ás etapas iniciais do proxecto que requirisen unha alta participación do cliente e de unha ou dúas semanas para os *sprints* adicados a tarefas de desenvolvemento, dependendo da importancia e entidade das tarefas que durante cada *sprint* se deben completar.
- Por cuestiós de organización decidiuse realizar unha soa reunión por *sprint*, na que se demostrasen e analizasen, en primeiro lugar, os requisitos completados do *sprint* finalizado (*Sprint Review*) e a continuación se planificasen o resto de actividades relativas ó novo *sprint*, xa comentadas no apartado 2.2. Aínda que *scrum* recomenda que durante a demostración de requisitos únicamente se revisen aqueles requisitos que foron completados para non crear falsas expectativas, neste caso tomouse a decisión de facer unha revisión completa do *sprint*, incluíndo as tarefas completadas e tamén as non rematadas, pois tratándose dunha ferramenta na que a interacción do usuario é fundamental, poderíase axilizar a detección e corrección de erros deste tipo.
- Outro aspecto que se debe mencionar con respecto á aplicación de *scrum* é o relativo á realización de reunións de sincronización diárias que, aínda sendo un dos fundamentos da metodoloxía, se decidiu non respectar debido a que o desenvolvemento ía ser levado a cabo de xeito individual. Porén, considerouse necesario que o equipo de dirección estivese dispoñible, preferiblemente de xeito presencial, durante todo o proxecto para aclarar ou resolver calquera conflito que pudera xurdir.
- A organización das diferentes tarefas do proxecto en *sprints* será detallada no apartado 3.8.

2.3. Planificación temporal

Para a elaboración do plan de traballo tentouse respectar, na medida do posible, a estrutura presentada no anteproxecto [25], adaptándoa á metodoloxía de desenvolvemento proposta. No referente ás estimacións temporais realizadas, considerouse unha xornada laboral de 25 horas semanais, como se especificaba tamén no citado documento.

En canto ó ciclo de vida, resulta difícil a definición dun conxunto de etapas independentes nas que organizar o proxecto, pois a metodoloxía elixida tende a agrupar temporalmente tarefas que en moitas ocasións se realizan en fases ben diferenciadas, como poden ser o análise e a implementación, e a facer que outras como o análisis de requisitos, que normalmente se levan a cabo durante as primeiras fases do ciclo de vida, poidan realizarse en paralelo con tarefas de deseño e implementación ó longo do proxecto, incluso cando o grao de avance no desenvolvemento é significativo. Tendo isto en conta, móstrase a continuación unha división conceptual do proxecto en fases, dispostas segundo a natureza das tarefas realizadas de xeito máis intensivo en cada unha delas.

2.3.1. Fase de inicio

Duración: 1 semana

Esta fase ten dous obxectivos principais. O primeiro deles é o de realizar unha descripción o máis ampla posible do produto, tratando de establecer os obxectivos e metas principais que se pretenden e a funcionalidade básica procurada. Neste punto terán lugar unha serie de reunións co cliente co propósito de comprender a súa necesidade e de que xeito podemos solventala. O segundo dos obxectivos, e non por iso menos importante, será a integración no novo entorno de traballo, e o establecemento dunhas canles de comunicación adecuadas e unha xerarquía de responsabilidade dentro do equipo de traballo.

2.3.2. Fase de elaboración

Duración: 2 semanas

Nesta etapa tratarase de propoñer unha descripción do sistema moi máis detallada, mediante un proceso de captura de requisitos formal, baseado en entrevistas e casos de uso, que nos permitirá propoñer unha primeira aproximación á arquitectura global do sistema, así como definir as probas que esta deberá de superar para ser aceptado. Unha vez feito isto, identificaranse as partes ou subsistemas nos que se pode dividir o producto para, en iteracións posteriores, realizar un deseño dos mesmos moi máis detallado e abordar a súa implementación.

Outra tarefa importante que deberá ser realizada nesta fase, é a análise e selección das tecnoloxías que van a ser utilizadas. Para este labor, proponse a realización dun estudo simple no que se avalíe a potencia e rendemento das diferentes alternativas, así como a creación dun pequeno prototipo do sistema que permita verificar a adecuación das opcións elixidas e garanta a viabilidade do sistema.

2.3.3. Fase de construción

Duración: 10 semanas

Neste fase procederase a realizar a implementación dos diferentes elementos que componen a arquitectura do sistema. Isto farase de forma progresiva en *sprints*, nos que se irán analizando,

deseñando e implementando un número limitado de requisitos xunto coas súas probas unitarias. Deste xeito, iranse obtendo periodicamente versións estables do produto final, aínda que limitadas en funcionalidade.

O producto resultante de cada iteración presentaráselle ó cliente para que este o avalíe, e deste xeito, farase un refinamento progresivo dos requisitos. Tratándose ademais, como é o caso, dunha ferramenta na que a interacción do usuario é esencial, poderá analizarse e deseñarse especialmente ben o xeito no que esta debería realizarse. Ó final desta fase obteremos un producto completo, integrado e totalmente funcional, listo para ser usado.

A organización dos diferentes *sprints* realizados durante esta fase será comentada máis adiante, cando sexan enunciados os requisitos e se obteña un conxunto inicial de tarefas que realizar.

2.3.4. Fase de transición

Duración: 2 semanas

Cando os diferentes compoñentes do producto estean rematados e as súas probas unitarias superadas, procederemos a realizar unha probas de máis alto nivel, tanto do sistema en si como de usabilidade e interacción con usuarios. Unha vez superadas estas, elaboraranse a documentación técnica da aplicación e o seu manual de usuario, e posteriormente, unha vez teñamos a aceptación do cliente, darase por rematado o desenvolvemento da ferramenta.

2.3.5. Fase de conclusión

Duración: 2 semanas

Unha vez rematado o producto, confeccionaranse a memoria e a presentación que permitirán a defensa do proxecto ante o tribunal, a partires dos documentos que se foron obtendo durante o resto de fases do desenvolvemento.

O diagrama de Gantt do proxecto (figura 2.5) ilustra como foron programadas as diferentes etapas do desenvolvemento en base á metodoloxía adoitada. Na figura 2.6 móstrase a EDT, co conxunto de paquetes de traballo que foi necesario realizar durante á execución das diferentes fases do ciclo de vida do proxecto.

2.4. Xestión da configuración

Un elemento de configuración é calquera produto de traballo, tanto produto final como produtos intermedios e tanto produtos entregables ó cliente como produtos internos do proxecto, dos cales o seu cambio pode resultar crítico para o correcto desenvolvemento do proxecto. A xestión da configuración é o proceso encargado de establecer os mecanismos axeitados que permitan manter a integridade de tódolos elementos de traballo, e entre os beneficios más salientables da implementación deste proceso están o aseguramento da correcta configuración do software, a capacidade de controlar cambios e a diminución dos sobreesforzos causados por problemas de integridade.

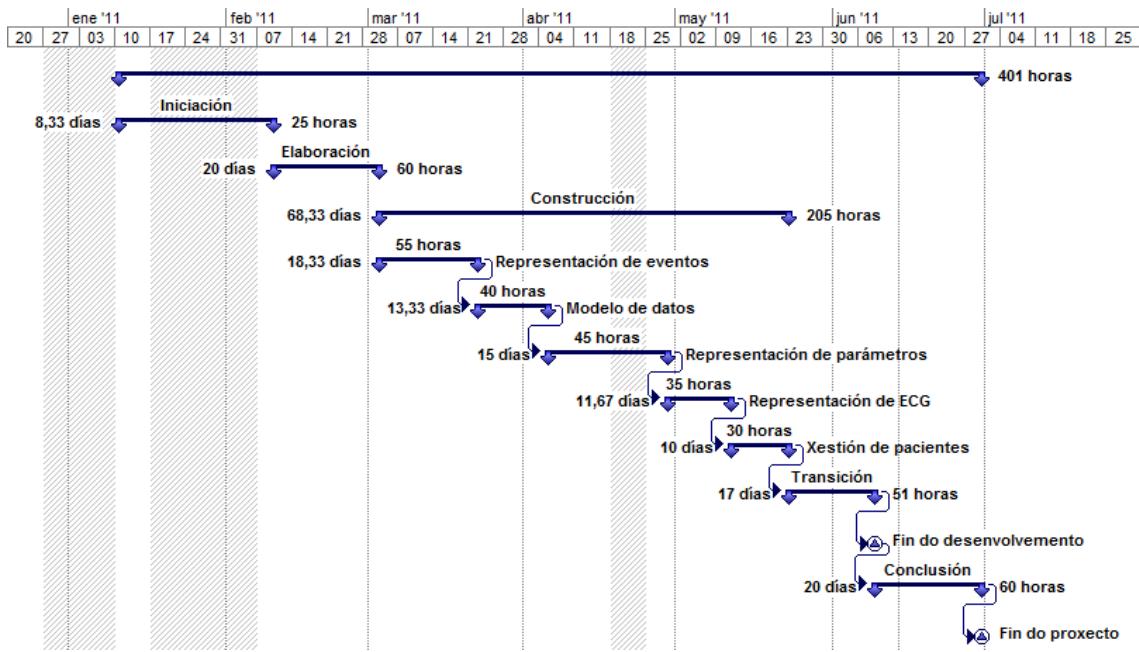


Figura 2.5: Diagrama de gantt do proxecto

Para establecer un plan de xestión da configuración debemos identificar os elementos que van ser controlados e as ferramentas e técnicas que se van usar para adquirir e xestionar ditos elementos. No noso caso os elementos que estarán baixo este proceso serán, por unha banda, o código fonte, e por outra, a documentación, tanto do código fonte coma do proxecto. A continuación detallaremos a estratexia que se seguirá en cada caso.

2.4.1. Control do código fonte

Para o control do código fonte usarase o software de control de versións *Subversion*, que permite unha xestión de cambios transparente, e o mantemento dun sistema de versionado que possibilita a recuperación do estado de calquera elemento da configuración nun instante temporal concreto. O repositorio no que se almacenará fisicamente o código estará situado nun servidor do Grupo de Sistemas Intelixentes do Departamento de Electrónica e Computación da USC.

2.4.2. Control da documentación

Como é típico cando se traballa dentro dun marco de desenvolvemento áxil, a maior parte da documentación técnica xérase a partires de comentarios no código fonte, gracias a ferramentas existentes nas diferentes entornas de desenvolvemento. Esta documentación almacenarase no mesmo repositorio có código ó que acompaña para garantir a súa consistencia.

Existen outro tipo de documentos que se xeran durante o desenvolvemento (documentos de deseño, diagramas de clases, etc.), e que non son susceptibles de ser sometidos a cambios con

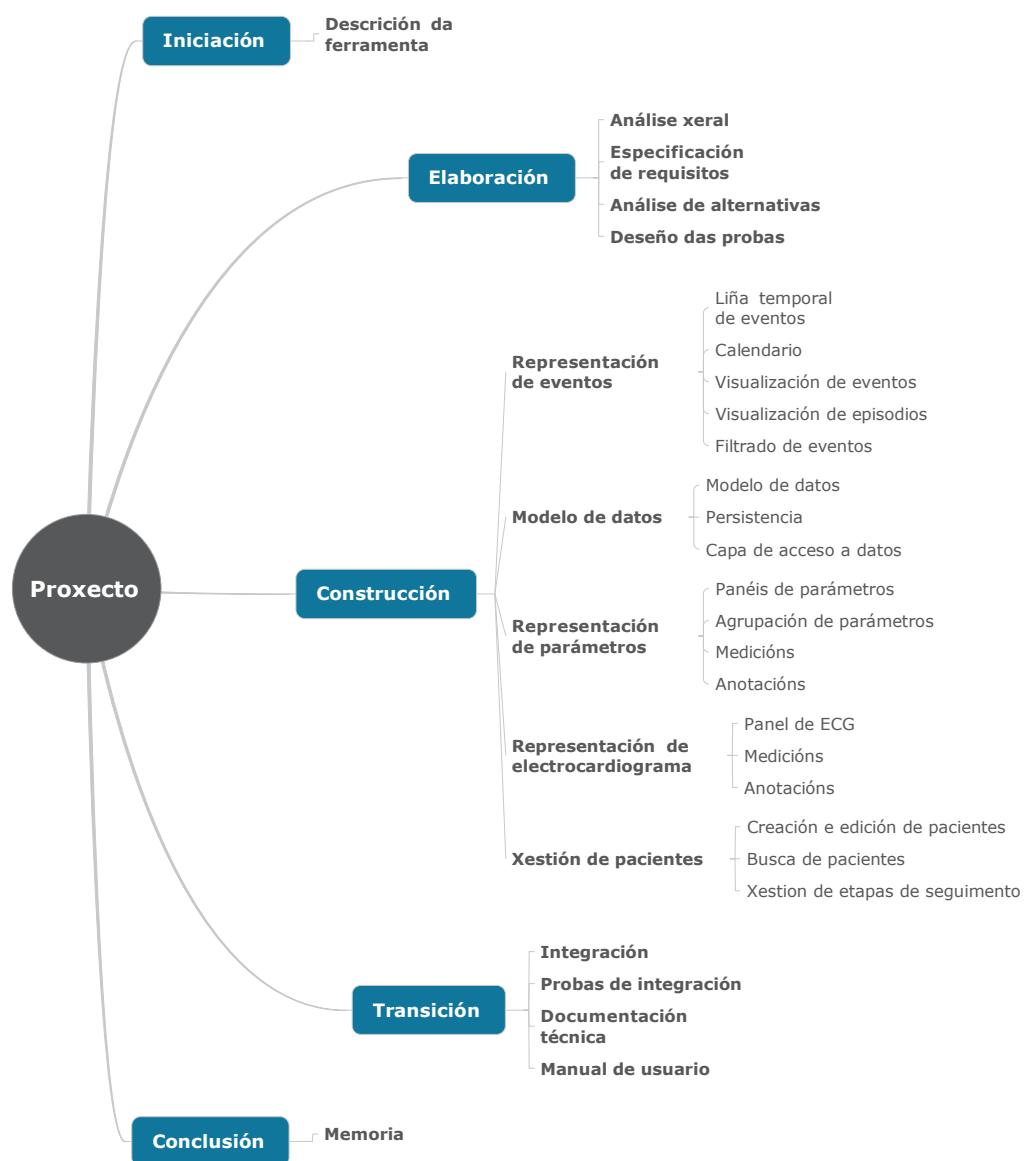


Figura 2.6: Estrutura de descomposición do traballo (EDT) do proxecto

tanta frecuencia coma o código fonte. Estes serán almacenados noutro repositorio, e xestionaranse mediante o software *Dropbox*, que nos protexe ante posibles perdidas de datos e nos permite realizar un control de versións simple, pero suficientemente potente para a tarefa en cuestión.

2.5. Análise de custos

Na táboa 2.1 móstrase unha relación dos medios necesarios para levar a cabo o proxecto. Durante a selección das librarías que se necesitaron durante o desenvolvemento tratouse de optar, sempre que foi posible, por ferramentas de uso libre e gratuíto que abaratasesen os custos do proxecto. Consideráronse, ademais, uns custos indirectos do 20 % referentes a outros gastos como electricidade, climatización, etc.

Ítem	Cantidad	Custo unitario	Custo Total
PC de Sobremesa	1	1.000,00 €	1000,00 €
Custo de persoal	412,5 horas	15,00 €	6.187,00 €
Subtotal:			7.187,50 €
Custos Indirectos:			+20 %
Total:			7.906,25 €

Táboa 2.1: Táboa de custos do proxecto.

Capítulo 3

Análise de requisitos

En calquera metodoloxía de desenvolvemento, o primeiro paso para a construción dun novo software é a especificación dos requisitos que deberá cumplir. No noso caso, comezamos por unha definición do conxunto de casos de uso da plataforma, a partir do cal se obtiveron despois dunha etapa de refinamento unha serie de requisitos funcionais, non funcionais, de interface e de calidade a satisfacer polo software implementado.

Comezaremos este capítulo cunha sección de definicións, na cal explicaremos de xeito preciso os diferentes conceptos específicos do dominio de aplicación ós que se fará referencia na descripción de requisitos e casos de uso. A continuación expoñeranse os diferentes casos de uso, posteriormente as restriccións de deseño ás que estará sometido o sistema, e finalmente o conxunto de requisitos de interfaces, comunicacións, funcionais, e de calidade, relacionando cada un deles cos casos de uso correspondentes. O esquema empregado para a obtención e especificación dos requisitos baséase no estándar IEEE 830-1998 [20].

3.1. Definicións

Evento

Suceso instantáneo relativo ó seguimento ou referente a accións realizadas polo usuario e de interese para o persoal clínico. A aplicación manexará diferentes tipos de evento, aínda que todos eles deberán levar asociados unha *referencia temporal* que indique o instante no que se produciron.

Exemplos de eventos poderían ser os seguintes:

- *Administración terapéutica ás 13:01:26 do día 1 de Xaneiro de 2011*
- *Medición da tensión arterial ás 13:01:26 do día 1 de Xaneiro de 2011*
- *Desconexión do sistema ás 13:01:26 do día 1 de Xaneiro de 2011*

Episodio

Suceso ou acción que se estende durante un lapso de tempo determinado, caracterizado por

unha referencia temporal de inicio, unha de finalización, e unha duración. Ademais destes atributos, cada episodio levará asociado un conxunto de atributos propios que o caracterizan en función do seu tipo. Un exemplo de episodio podería ser o seguinte:

- *Episodio de isquemia de 12:06 minutos de duración ás 13:01:26 do día 1 de Xaneiro de 2011, detectado sobre o parámetro de Desviación de ST*

Parámetro fisiolóxico

Variable física resultado dun proceso de medida realizado sobre o organismo do paciente; pode ser adquirido mediante algún dispositivo sensor, e posteriormente procesado e analizado co obxectivo de interpretar o estado e evolución do paciente. Exemplos de parámetros fisiológicos son o peso, a temperatura, ou a frecuencia cardiaca.

Electrocardiograma (ECG)

O electrocardiograma (ECG) é unha técnica non invasiva que permite rexistrar a actividade eléctrica do corazón mediante electrodos situados sobre o corpo do paciente. Un ECG estándar está formado por 12 derivacións ou canles, que se obteñen a partir das diferentes combinacións de electrodos para medir as distinas sinais procedentes do corazón. O trazado típico do ECG rexistrando un latido cardiaco normal consiste nunha onda P, un complexo QRS e unha onda T, os cales teñen asociados uns intervalos de duración e amplitude que se consideran normais, o seu estudio permite determinar a existencia de posibles anomalías na actividad do corazón. Actualmente o ECG é a ferramenta principal para o diagnóstico de enfermedades cardiovasculares.

Familia morfolóxica

Unha familia morfolólica de ECG é un grupo de latidos con morfoloxía semellante; pode representarse como un latido que resume a morfoloxía do conxunto de latidos que a forma. Permite resumir os diferentes tipos de latidos que tiveron lugar durante a monitorización do paciente, facilitando a identificación rápida da existencia de anomalías.

Diagrama de tendencias

Conxunto de series temporais que mostra a evolución de parámetros fisiológicos de interese. No presente traballo, o diagrama de tendencias reúne un conxunto de parámetros resultado de mediciones realizadas sobre o ECG do paciente: intervalo RR, segmento PQ, etc.

Saturación de oxíxeno

Concentración de oxihemoglobina en sangue medida en tanto por cento no sistema vascular periférico. Na súa medición emprégase polo xeral unha pinza de pulsioximetría, que colócase nalgún dos dedos das mans do paciente.

Variabilidade da frecuencia cardíaca (HRV)

Propiedade da variación temporal do ritmo cardiaco, resultado da manifestación antagónica dos sistemas nerviosos simpático e parasimpático. A análise da variabilidade da frecuencia cardiaca emprégase como estratificador de risco, asociándose unha limitada variabilidade cun deterioro na saúde do paciente.

3.2. Casos de uso

Os casos de uso son unha técnica de captura de información utilizada amplamente para describir de xeito informal a interacción típica entre os usuarios dun sistema (actores), e o propio sistema. Mostran, pois, o xeito no que o sistema traballa ou se desexa que traballe, sen atender a cuestións específicas de implementación ou comportamento. Do mesmo xeito, poden ser de axuda para validar a arquitectura e verificar o sistema durante o seu desenvolvemento.

A continuación farase unha descripción dos casos de uso identificados no sistema, cos seus correspondentes actores, segundo a estructura extraída de [14].

3.2.1. Descripción de actores

Para a interacción co sistema, foron identificados os seguintes actores:

- **Persoal clínico:** Este actor engloba ó conxunto de facultativos que podería fazer uso da aplicación. É o actor por defecto do sistema, e por comodidade, de agora en diante referirémonos a el simplemente como *usuario*.
- **Administrador:** Actor que levará a cabo tarefas de xestión e mantemento como a modificación de configuracións ou a administración de usuarios.

3.2.2. Descripción de casos de uso

A continuación móstrase o conxunto de casos de uso do sistema, que foron identificados mediante a realización de entrevistas co cliente. Na figura 3.1 móstrase o diagrama de casos de uso do sistema.

Caso de uso CU-01: Xestión de usuarios

Actores

Administrador

Escenario principal

O caso de uso comeza cando se desexa incorporar a un determinado asistente clínico ó conxunto de usuarios da aplicación.

Un administrador identifícase no sistema e diríxese a área de xestión de usuarios. Unha vez alí elixe dar de alta un novo usuario e introduce os datos pertinentes do asistente en cuestión, que este poderá modificar unha vez dado de alta, indica que se trata dun membro do persoal clínico, e o asistente queda rexistrado no sistema.

Escenario alternativo 1

O administrador equivócase introducindo os seus datos de acceso, co que o sistema non lle permite acceder e mostra un aviso informando do suceso.

Escenario alternativo 2

Precísase crear outro usuario de carácter administrativo. Do mesmo xeito que antes, introdúcense os seus datos, indicando agora que se trata dun administrador do sistema e o usuario queda dado de alta.

Caso de uso CU-02: Xestión de pacientes

Actores

Persoal clínico, Administrador

Escenario principal

O caso de uso comeza cando se quere iniciar o seguimento dun determinado paciente. Un asistente clínico ou un administrador (por claridade suporemos que un asistente clínico) identifícase no sistema introducindo as súas credenciais de acceso e diríxese á área de xestión de pacientes. Alí elixe a creación dun novo paciente, introduce os datos pertinentes, selecciona o conxunto de médicos que se lle asignarán, e configura unha nova etapa de seguimento, que dá comezo nunha determinada data. Unha vez realizados estes pasos o paciente pasa a formar parte da lista de pacientes dados de alta no sistema.

Escenario alternativo 1

O asistente equivócase introducindo os datos do paciente. Máis tarde búscalo e realiza as correccións necesarias.

Escenario alternativo 2

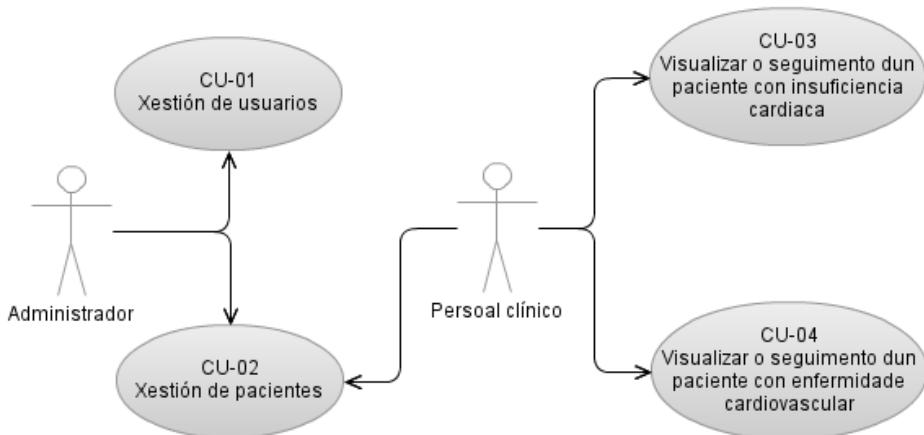


Figura 3.1: Diagrama de casos de uso do sistema

O asistente crea por erro dousas etapas de seguimento. Posteriormente accede ó perfil do paciente, elixe unha das etapas creadas e elimínaa. O sistema permiteo sempre e cando a etapa non conteña datos de seguimento.

Caso de uso CU-03: Visualizar o seguimento dun paciente con insuficiencia cardiaca**Actores**

Persoal clínico

Escenario principal

O caso de uso comeza cando un asistente clínico desexa visualizar o seguimento dun paciente con insuficiencia cardiaca. O seguimento deste tipo de paciente consiste nun conxunto de probas puntuais, realizadas cunha periodicidade diaria e aproximadamente á mesma hora.

O asistente clínico accede ó seguimento realizado nun determinado período seleccionando unha das etapas de seguimento dispoñibles no perfil do paciente. Unha vez feito isto, selecciona o día que deseja ver, e accede ás probas realizadas no mesmo, que se visualizarán como unha serie de eventos facilmente clasificables a simple vista. Se foron rexistrados un elevado número de eventos na data seleccionada, e resulta difícil a visualización dos mesmos, o asistente pode axustar a escala temporal de xeito que a visualización lle resulte máis cómoda e desprázase temporalmente para poder visualizar o resto de eventos.

A probas que se poderán visualizar consistirán nunha monitorización de electrocardiograma de ata 5 minutos de duración, unha medida de peso e de tensión arterial, e un cuestionario no que se indicará a evolución dos síntomas de fatiga e de edema (indicando se melloraron, empeoraron ou seguen igual có día anterior).

- Se se selecciona a monitorización de electrocardiograma, abrirase unha ferramenta que permita a visualización deste tipo de sinal. O asistente poderá a través dela seleccionar os canais de ECG que deseja ver, axustar as escalas de tempo e amplitude segundo precise, establecer diferentes cuadrículas que lle axuden a caracterizar o sinal, e realizar medicións sobre os diferentes canais amosados en caso de ser necesario.
- En canto ó resto de probas, dado que resultan nun conxunto de valores puntuais, poderá ver os dese día accedendo ós detalles de cada evento mostrado, ou unha evolución dos mesmos durante un intervalo de tempo de ata dousas semanas, para o cal se abrirá unha nova ventá que amosará eses valores graficamente. Poderá tamén seleccionar que tipo de valores se mostran, así como axustar o intervalo de tempo que se está a visualizar segundo considere, e desprazarse polos valores rexistrados durante o resto da etapa de xeito simple.

Caso de uso CU-04: Visualizar o seguimento dun paciente con enfermidade cardiovascular**Actores**

Persoal clínico

Escenario principal

O caso de uso comeza cando un asistente clínico desexa visualizar o seguimento dun paciente con enfermidade cardiovascular. O seguimento deste tipo de paciente realiza de xeito continuado durante a noite, mediante unha monitorización de electrocardiograma e saturación de osíxeno.

O asistente clínico poderá, para cada día, visualizar nun panel, de xeito individual ou por grupos, os diferentes parámetros que componen o diagrama de tendencias obtido a partir do sinal de electrocardiograma, permitíndoselle seleccionar que parámetros desexa ver, e modificar a súa escala temporal e de amplitude. O sinal de saturación de osíxeno poderá visualizarse conjuntamente con estes parámetros.

Sobre cada parámetro visualizado mostraranse marcas indicando episodios que foron detectados sobre el durante a monitorización, e no caso de estes fagan referencia a fragmentos de electrocardiograma mostrarse unha marca indicándoo, que ademais permita acceder á visualización do fragmento. Os diferentes episodios mostraranse tamén xunto cos eventos do día, e a través deles poderase acceder á visualización do parámetro sobre o que foron detectados. Permitirse ademais a adición de anotacións sobre puntos concretos ou intervalos do sinal, e a realización de medicións de tempo e amplitude sobre o mesmo.

O asistente clínico poderá acceder tamén á ventá de ECG para realizar unha solicitude de electrocardiograma en tempo real ó dispositivo de seguimento remoto, en fragmentos que poderán ter ata un minuto de duración. Por outra banda, permitiráselle que desde esa mesma ventá poida visualizar a información de familias morfolóxicas obtidas a partir do ECG, e acceder ó último fragmento de sinal sobre o que se detectou unha determinada familia.

Por último, o asistente poderá visualizar a información sobre variabilidade da frecuencia cardiaca, que se obterá periodicamente cada hora de monitorización.

3.3. Restriccóns de deseño

Requisito RD.1

Título: Implementación en Java dos módulos de persistencia

Descripción: Os módulos de persistencia serán implementados en Java, pois debemos fazer uso de interfaces xa implementadas nesta linguaxe para comunicarnos con Servando.

Importancia: Esencial

Requisito RD.2

Título: Utilización de librarías non privativas na medida do posible

Descripción: Sempre que precisemos fazer uso de librarías externas, e na medida en que sexa posible, tratarase de fazer uso de librarías de carácter libre e gratuíto.

Importancia: Condicional

3.4. Requisitos funcionais

Requisito FN.1

Título: Alta de usuarios

Descripción: A aplicación debe permitir que se creen novas contas de usuario, tanto para o persoal clínico como para os administradores do sistema.

Casos de uso relacionados: CU-01

Importancia: Esencial

Requisito FN.2

Título: Baixa de usuarios

Descripción: A aplicación debe permitir que se borren contas de usuario do sistema.

Casos de uso relacionados: CU-01

Importancia: Esencial

Requisito FN.3

Título: Modificación de usuarios

Descripción: A aplicación debe permitir que se modifiquen contas de usuario existentes.

Casos de uso relacionados: CU-01

Importancia: Opcional

Requisito FN.4

Título: Identificación de usuarios

Descripción: A aplicación debe permitir que os usuarios dados de alta se identifiquen no sistema mediante nome de usuario e contrasinal.

Casos de uso relacionados: CU-01, CU-02, CU-03, CU-04

Importancia: Esencial

Requisito FN.5

Título: Xestión de perfiles de usuario

Descripción: A aplicación de permitir a cada usuario a configuración de determinados parámetros do sistema, tales como colocación das ventás, segundo as súas preferencias.

Casos de uso relacionados: CU-01

Importancia: Opcional

Requisito FN.6

Título: Alta de pacientes

Descripción: A aplicación debe permitir que se dean de alta novos pacientes.

Casos de uso relacionados: CU-02

Importancia: Esencial

Requisito FN.7

Título: Baixa de pacientes

Descripción: A aplicación debe permitir que se dean de baixa pacientes dados de alta.

Casos de uso relacionados: CU-02

Importancia: Opcional

Requisito FN.8

Título: Edición de perfís de pacientes

Descripción: A aplicación de permitir que se edite o perfil dun paciente dados de alta no sistema. Enténdese por edición do perfil a posibilidade de modificar os seus datos persoais.

Casos de uso relacionados: CU-02

Importancia: Opcional

Requisito FN.9

Título: Consulta de pacientes

Descripción: A aplicación debe permitir consultar os pacientes dados alta no sistema, así como a realización de buscas que permitan filtralos en base ós seus datos persoais.

Casos de uso relacionados: CU-02

Importancia: Esencial

Requisito FN.10

Título: Asignación de médicos a pacientes

Descripción: A aplicación debe permitir que, dende o perfil dun paciente, se lle poidan asignar a este unha serie de médicos (usuarios dados de alta no sistema) que se ocupen do seu seguimento.

Casos de uso relacionados: CU-02

Importancia: Esencial

Requisito FN.11

Título: Edición de etapas de seguimiento

Descripción: A aplicación debe permitir, para un paciente dado, a creación novas etapas de seguimento e o seu posterior borrado, en caso de que non existan datos de monitorización sobre as mesmas.

Casos de uso relacionados: CU-02

Importancia: Esencial

Requisito FN.12

Título: Consulta retrospectiva de pacientes

Descripción: A aplicación deberá permitir a visualización do seguimento realizado a pacientes que xa foron dados de alta.

Casos de uso relacionados: CU-02, CU-03, CU-04

Importancia: Esencial

Requisito FN.13

Título: Consulta en tempo real de pacientes

Descripción: A aplicación deberá permitir a visualización en tempo real de pacientes que actualmente están baixo seguimento domiciliario.

Casos de uso relacionados: CU-02, CU-03, CU-04

Importancia: Esencial

Requisito FN.14

Título: Calendario

Descripción: A interface da aplicación deberá contar cun calendario que permita a visualizar o seguimento realizado a un determinado paciente nunha data seleccionada.

Casos de uso relacionados: CU-03, CU-04

Importancia: Esencial

Requisito FN.15

Título: Sinalización de eventos no calendario

Descripción: O calendario da aplicación deberá contar cun sistema de previsualización de eventos, de xeito que se poidan identificar facilmente aqueles días con eventos significativos.

Casos de uso relacionados: CU-03, CU-04

Importancia: Opcional

Requisito FN.16

Título: Liña de representación temporal de eventos

Descripción: A aplicación deberá contar cunha liña de representación temporal de eventos que permita extraer de xeito visual, polo menos, o tipo de evento do que se trata e o instante no que se produciu. Deberase permitir tamén a navegación temporal cara adiante e cara atrás e a selección entre diferentes niveis de granularidade temporal (horas, días, semanas, etc.).

Casos de uso relacionados: CU-03, CU-04

Importancia: Esencial

Requisito FN.17

Título: Visualización dos detalles dun evento

Descripción: A aplicación deberá mostrar os detalles dun determinado evento (tipo de evento, descripción, data, etc.) nun cadro emergente cando o usuario o seleccione.

Casos de uso relacionados: CU-03, CU-04

Importancia: Esencial

Requisito FN.18

Título: Visualización dos detalles dun episodio

Descripción: Ao seleccionar un episodio na liña temporal, a aplicación deberá mostrar na área de visualización de parámetros o parámetro sobre o que se detectou o episodio en cuestión.

Casos de uso relacionados: CU-04

Importancia: Esencial

Requisito FN.19

Título: Franxa de sincronización temporal

Descripción: A aplicación deberá mostrar, na liña de representación de eventos, unha franxa que indique o intervalo que se está a visualizar na área de representación de parámetros, no caso de haxa parámetros activos.

Casos de uso relacionados: CU-03, CU-04

Importancia: Opcional

Requisito FN.20

Título: Filtro de eventos representados

Descripción: A aplicación deberá permitir que o usuario filtre os eventos que se mostran na liña de representación temporal de eventos segundo o seu tipo.

Casos de uso relacionados: CU-13

Importancia: Opcional

Requisito FN.21

Título: Selección de parámetros

Descripción: A aplicación deberá mostrar os diferentes parámetros dispoñibles para visualizar e permitir que o usuario seleccione os que desexa ver en cada momento.

Casos de uso relacionados: CU-03, CU-04

Importancia: Esencial

Requisito FN.22

Título: Agrupación de parámetros

Descripción: A aplicación deberá permitir que o usuario visualice diferentes grupos de parámetros, podendo este seleccionar os parámetros que desexa visualizar en cada grupo.

Casos de uso relacionados: CU-03, CU-04

Importancia: Esencial

Requisito FN.23

Título: Representación de diagramas de tendencia.

Descripción: A aplicación deberá permitir a representación dos sinais dos diferentes parámetros do diagrama de tendencias que se obtén a partir do sinal de electrocardiograma. Deberase aproveitar sempre todo o espacio dispoñible, independentemente do número de parámetros visualizados, co fin de mellorar a experiencia do usuario.

Casos de uso relacionados: CU-03, CU-04

Importancia: Esencial

Requisito FN.24

Título: Edición de anotacións

Descripción: A aplicación deberá permitir a adición e eliminación de anotacións textuais sobre puntos ou intervalos do sinal de cada parámetro.

Casos de uso relacionados: CU-04

Importancia: Opcional

Requisito FN.25

Título: Mostrar/ocultar anotacións

Descripción: A aplicación deberá permitir que o usuario active ou desactive a visualización das anotacións realizadas sobre cada parámetro visualizado.

Casos de uso relacionados: CU-04

Importancia: Opcional

Requisito FN.26

Título: Representación de parámetros fisiológicos

Descripción: A aplicación deberá permitir a representación de sinais de parámetros fisiológicos como a tensión arterial ou o peso. Dado que a frecuencia coa que se obteñen este tipo de parámetros é moito menor cá frecuencia ca que se obteñen os parámetros de electrocardiograma (un dato por día fronte a catro datos por segundo) non se permitirá a visualización conxunta de parámetros de ámbolos dous grupos.

Casos de uso relacionados: CU-03, CU-04

Importancia: Esencial

Requisito FN.27

Título: Desprazamento temporal na área de parámetros

Descripción: A aplicación deberá permitir que o usuario se desprace temporalmente polo sinal dos diferentes parámetros visualizados, tanto para parámetros fisiológicos coma para os diagramas de tendencia.

Casos de uso relacionados: CU-03, CU-04

Importancia: Esencial

Requisito FN.28

Título: Área de representación de electrocardiograma

Descripción: A aplicación deberá permitir a visualización de sinais de electrocardiograma, pondo o usuario seleccionar as canles que deseja ver en cada momento.

Casos de uso relacionados: CU-03, CU-04

Importancia: Esencial

Requisito FN.29

Título: Ventá de representación de familias morfolóxicas

Descripción: A aplicación deberá permitir a visualización das familias morfolóxicas detectadas sobre o electrocardiograma nun instante determinado. Ademais deberase permitir o acceso ó último fragmento de sinal sobre o que se detectou o último latido dunha familia concreta.

Casos de uso relacionados: CU-04

Importancia: Esencial

Requisito FN.30

Título: Sincronización temporal de ventás

Descripción: A aplicación deberá manter sincronizadas as distintas ventás de representación (incluíndo o calendario) de xeito que todas elas avancen e retrocedan convxuntamente no tempo, independentemente de con cal está a traballar o usuario.

Casos de uso relacionados: CU-03, CU-04

Importancia: Esencial

Requisito FN.31

Título: Visualización de cuadrículas

Descripción: A aplicación deberá permitir a selección entre un conxunto de cuadrículas predefinidas durante a visualización de parámetros e de electrocardiograma. No caso da visualización de parámetros, posto que os parámetros obtidos poden ser medidos en diferentes unidades e tomar valores en diferentes rangos, permitirase a configuración de cuadrículas independentes para a amplitude de cada un deles.

Casos de uso relacionados: CU-03, CU-04

Importancia: Esencial

Requisito FN.32

Título: Manipulación de escalas de amplitud

Descripción: A aplicación deberá permitir a manipulación de escalas de amplitud durante a visualización de parámetros e fragmentos de ECG. Coma no caso das cuadrículas a selección da escala de amplitud farase de xeito independente para cada parámetro.

Casos de uso relacionados: CU-03, CU-04

Importancia: Esencial

Requisito FN.33

Título: Manipulación de escalas temporais

Descripción: A aplicación deberá permitir a manipulación de escalas temporais durante a visualización de eventos, parámetros e fragmentos de ECG. No caso da visualización de ECG e dos parámetros que del se derivan as escalas máximas permitidas serán da orde dos minutos, mentres que na visualización de parámetros fisiológicos e na liña temporal de eventos se permitirán escalas máximas de ata 15 días.

Casos de uso relacionados: CU-03, CU-04

Importancia: Esencial

Requisito FN.34

Título: Realización de medicións

Descripción: As áreas de representación de parámetros e de electrocardiograma deberán permitir a realización de medicións sobre os diferentes sinais visualizados.

Casos de uso relacionados: CU-03, CU-04

Importancia: Esencial

Requisito FN.35

Título: Área de representación da variabilidad de frecuencia cardiaca

Descripción: A aplicación deberá permitir a visualización da información sobre a variabilidade da frecuencia cardiaca, que se obtén durante as monitorizacións de electrocardiograma cunha periodicidade dunha hora. Ademais da información de HRV para cada hora, tamén se deberá amosar a evolución da mesma en periodos de ata 8 horas.

Casos de uso relacionados: CU-03, CU-04

Importancia: Opcional

3.5. Requisitos de proxecto

Requisito PR.1

Título: Data límite 11 de xullo

Descripción: A data límite de entrega do proxecto será o 11 de Xullo de 2011, data establecida no regulamento de traballos fin de grao para GREI da USC.

Importancia: Esencial

3.6. Requisitos de calidade

Requisito CA.1

Título: Alto rendemento

Descripción: A aplicación deberá de xestionar de xeito eficiente e transparente ó usuario o alto volume de datos co que se traballara nas gráficas.

Importancia: Esencial

Requisito CA.2

Título: Interfaz usable e ergonómica

Descripción: A interacción entre o usuario e o sistema deberá ser fluída, de xeito que este poida realizar as súas tarefas de xeito eficiente e co menor traballo posible. Tratarase tamén, de proporcionar atallos de teclado que permitan a realización da maior parte das tarefas dun xeito máis rápido e cómodo.

Importancia: Esencial

Requisito CA.3

Título: Axuda

Descripción: A aplicación deberá contar cunha axuda que facilite ó usuario o manexo da ferramenta. Á marxe disto, farase énfase no uso de metáforas visuais que axuden ao persoal clínico a comprender e interpretar rapidamente a información que deseñe visualizar, e o guén durante a interacción co sistema.

Importancia: Esencial

Requisito CA.4

Título: Internacionalización

Descripción: Deberase deseñar a aplicación de xeito que poida adaptarse a diferentes idiomas e rexións sen necesidade de realizar cambios no deseño ou no código.

Importancia: Opcional

3.7. Requisitos de almacenamiento

Requisito AL.1

Título: Automatización da persistencia

Descripción: A aplicación deberá contar cun sistema de mapeo obxecto-relacional que permita, a través dun determinado modelo de obxectos, xerar de xeito automático o modelo relacional asociado, permitindo así unha extensión simple do modelo datos.

Importancia: Esencial

3.8. Organización de *sprints*

Segundo *scrum*, unha vez expostos e priorizados os requisitos do proxecto debemos abordar a súa implementación de xeito incremental, mediante *sprints*. En cada *sprint* implementaranse unha serie de requisitos de acordo coa prioridade que teñen nese momento para o cliente e para o desenvolvemento do proxecto. A continuación exponse o conxunto de *sprints* que foi necesario realizar durante as diferentes fases do proxecto. Para os correspondentes á fase de desenvolvemento indicaranse os requisitos que foron completados en cada un deles. Hai que destacar que para a planificación dos diferentes *sprints* se tratou de minimizar o número de obxectivos/requisitos nos que traballar simultaneamente, co fin de aumentar a capacidade de reacción ante cambios e situacións inesperadas.

Identificador:	Sprint-01
Nome:	Descripción do producto
Fase:	Inicio
Comezo:	10/01/2011
Fin:	13/01/2011
Descripción do sprint:	Definición de obxectivos, e caracterización do producto.

Identificador:	Sprint-02
Nome:	Análise de requisitos
Fase:	Elaboración
Comezo:	14/01/2011
Fin:	19/01/2011
Descripción do sprint:	Análise do sistema e especificación dos requisitos iniciais do producto.

Identificador:	Sprint-03
Nome:	Selección de tecnoloxías
Fase:	Elaboración
Fase:	Inicio
Comezo:	20/01/2011
Fin:	26/01/2011
Descripción do sprint:	Análise de alternativas e selección de ferramentas e tecnoloxías.

Identificador:	Sprint-04
Nome:	Liña temporal de eventos
Fase:	Construcción
Comezo:	27/01/2011
Fin:	02/02/2011
Descripción do sprint:	Realización das tarefas relativas a representación de eventos mediante unha liña temporal.
Requisitos realizados:	FN.15, FN.16, FN.17, FN.20

Identificador:	Sprint-05
Nome:	Persistencia
Fase:	Construcción
Comezo:	03/01/2011
Fin:	09/02/2011
Descripción do sprint:	Elaboración do modelo de datos e do motor de persistencia.
Requisitos realizados:	AL.1

Identificador:	Sprint-06
Nome:	Representación de parámetros
Fase:	Construcción
Comezo:	10/02/2011
Fin:	23/02/2011
Descripción do sprint:	Creación dos componentes de representación e agrupación visual de parámetros
Requisitos realizados:	FN.22, FN.23

Identificador:	Sprint-07
Nome:	Representación de parámetros (II)
Fase:	Construcción
Comezo:	24/02/2011
Fin:	03/03/2011
Descripción do sprint:	Creación dos mecanismos de selección de parámetros. Desprazamento temporal na área de parámetros e edición de anotacións.
Requisitos realizados:	FN.24, FN.25, FN.27, FN.30
Identificador:	Sprint-08
Nome:	Representación de episodios
Fase:	Construcción
Comezo:	04/03/2011
Fin:	17/03/2011
Descripción do sprint:	Representación de episodios na liña temporal, visualización dos detalles dos mesmos, e sincronización de ventás.
Requisitos realizados:	FN.18, FN.19
Identificador:	Sprint-09
Nome:	Representación de ECG
Fase:	Construcción
Comezo:	19/04/2011
Fin:	25/04/2011
Descripción do sprint:	Realización da ventá de representación de electrocardiograma, selección de canles a visualizar, e representación do sinal de ECG.
Requisitos realizados:	FN.28
Identificador:	Sprint-10
Nome:	Representación de ECG (II)
Fase:	Construcción
Comezo:	25/04/2011
Fin:	01/05/2011
Descripción do sprint:	Representación de familias morfológicas.
Requisitos realizados:	FN.29

Identificador:	Sprint-11
Nome:	Xestión de escalas e cuadrículas
Fase:	Construcción
Comezo:	02/05/2011
Fin:	10/05/2011
Descripción do sprint:	Xestión de cuadrículas e escalas nas ventás de representación de parámetros e electrocardiograma.
Requisitos realizados:	FN.31, FN.32, FN.33, FN.34
Identificador:	Sprint-12
Nome:	Xestión de pacientes
Fase:	Construcción
Comezo:	11/05/2011
Fin:	17/05/2011
Descripción do sprint:	Realización das tarefas relativas á creación e edición de pacientes.
Requisitos realizados:	FN.6, FN.7, FN.8, FN.9, FN.10
Identificador:	Sprint-13
Nome:	Xestión de etapas de seguimiento - Xestión de usuarios
Fase:	Construcción
Comezo:	18/05/2011
Fin:	24/05/2011
Descripción do sprint:	Implementación da lóxica de xestión de etapas de seguimiento e da xestión de usuarios.
Requisitos realizados:	FN.1, FN.2, FN.4, FN.11, FN.12, FN.13
Identificador:	Sprint-14
Nome:	Representación de parámetros fisiológicos
Fase:	Construcción
Comezo:	25/05/2011
Fin:	01/06/2011
Descripción do sprint:	Selección de parámetros fisiológicos e representación dos mesmos.
Requisitos realizados:	FN.26
Identificador:	Sprint-15
Nome:	Probas e integración
Fase:	Transición
Comezo:	02/06/2011
Fin:	08/06/2011
Descripción do sprint:	Realización de probas do sistema e integración.

Identificador:	Sprint-16
Nome:	Manual de usuario
Fase:	Transición
Comezo:	09/06/2011
Fin:	15/06/2011
Descripción do sprint:	Elaboración da documentación técnica da aplicación e do manual de usuario.

Identificador:	Sprint-17
Nome:	Memoria
Fase:	Conclusión
Comezo:	16/06/2011
Fin:	02/07/2011
Descripción do sprint:	Elaboración da memoria do proxecto.

Hai que subliñar que non se pode entender o conxunto de *sprints* expostos como unha planificación, senón como un resultado da aplicación dunha metodoloxía, neste caso *Scrum*, ó conxunto de tarefas necesarias para a realización do proxecto, das que algúnhas estaban claras dende o inicio do proxecto, e outras foron emerxendo durante a súa evolución.

A duración dos *sprints* realizados oscila entre unha e dúas semanas, como se decidiu ó comezo do proxecto. A razón principal que fai que en ocasións a duración dos diferentes sprints varíe é a dificultade de reunirse co cliente o último día do *sprint*.

O traballo realizado durante a elaboración do proxecto sitúase en torno ás 450 horas, das que a maior parte (307 horas) pertencen á parte de desenvolvemento, isto é, ás fases de elaboración e construcción mostradas no diagrama de gantt da figura 2.5, durante as que se realizaron un total de 127 tarefas. Durante a planificación inicial, e tendo en conta o traballo que se debe realizar durante un proxecto fin de grao, estimouse para a execución desas dúas fases se disporía dunhas 270 horas. Existe, pois, entre o traballo planificado e o traballo realizado un desfase de arredor de 30 horas, que non se considera crítico, dada a inexperiencia en tarefas de xestión e a dificultade para realizar estimacións precisas. Ademais, en ningún caso este desfase puido supoñer a entrega fóra de prazo do proxecto ou poñer en perigo a correcta finalización do mesmo.

Capítulo 4

Arquitectura, tecnoloxías e ferramentas

Neste capítulo partirse do conxunto de casos de uso do sistema e dos requisitos expostos para definir a arquitectura do sistema. En primeiro lugar, avaliaranse as diferentes opcións existentes para a definición da arquitectura de máis alto nivel do sistema, para a continuación expoñer a solución proposta e describir o conxunto de componentes que a conforman. Unha vez comentada a arquitectura proposta, procederase á discusión das diferentes ferramentas e tecnoloxías necesarias para o seu deseño e implementación, valorando as diferentes alternativas availables sempre que sexa necesario.

4.1. Alternativas dispoñibles

Unha vez se teñen analizado os diferentes casos de uso identificados, e definido os requisitos do proxecto, é o momento de deseñar a arquitectura global do sistema. Para a súa definición é importante analizar, ademais dos requisitos e das necesidades actuais do cliente, as posibles melloras ou modificacións que se poidan realizar posteriormente, para deste xeito propoñer unha arquitectura robusta e scalable que permita a evolución futura do sistema. Neste caso resulta fácil, á vista dos obxectivos expostos e dos casos de uso identificados, adiantar posibles melloras do sistema (un servizo de citas para os pacientes dados de alta, a posibilidade de que esas citas se puidesen realizar de xeito remoto do mesmo xeito que se realiza o seguimento, ou un sistema de seguimento xeográfico que permita coñecer a localización dun paciente durante un suceso concreto), que serían más ou menos difíciles de realizar dependendo da arquitectura que agora se propoña.

Na descripción técnica incluída no anteproxecto [25] faise unha primeira aproximación á arquitectura do sistema. Como se explica en dito documento, pártese de que xa existe un servidor de aplicacións no que se executa o “sistema xestor de dispositivos de seguimento” de Servando, o cal se ocupa de recibir os datos dende os dispositivos móbiles de seguimento domiciliario. Para almacenar os datos proponse a creación un motor de persistencia que proporcione a funcionalidade necesaria

para realizar operacións de almacenamento, extracción, actualización e borrado e sirva como base para estender o modelo facilmente. Por outra banda, proponse a creación dunha aplicación cliente que permita levar a cabo todas as tarefas relacionadas coa visualización do seguimento e a xestión de pacientes. En canto á creación do motor de persistencia, implementarase en Java para cumplir cas interfaces existentes no sistema e permitir que sexa utilizado polo sistema xestor de dispositivos de Servando, e executarase xunto con el no mesmo servidor de aplicacións. Polo tanto, as alternativas que se barallarán en canto á arquitectura do sistema serán referentes á parte do cliente, que será o que se ocupe das tarefas de visualización.

Débese ter en conta que a arquitectura proposta restrinxirá de xeito notable o conxunto de tecnoloxías e ferramentas que será posible utilizar, polo que é conveniente antes de propoñer unha arquitectura concreta, analizar de xeito global as características e necesidades do producto, de xeito que non descartemos prematuramente tecnoloxías ou ferramentas que nos poden facilitar o desenvolvemento enormemente, non permitindo en ningún momento que sexa a utilización dunha determinada tecnoloxía a que condicione o deseño da arquitectura. Isto é especialmente importante neste proxecto, pois unha destas necesidades é a de contar con tecnoloxías que nos permitan construír compoñentes visuais altamente interactivos e ó mesmo tempo atractivos para o usuario, que respondan ás necesidades de visualización extraídas dos casos de uso e dos requisitos.

Chegados a este punto, barállanse principalmente dúas alternativas para a construcción da aplicación cliente, pois proponse por unha banda a implementación do mesmo como unha aplicación de escritorio tradicional, e por outra a definición dunha arquitectura RIA (Rich Internet Application) [11] na que a aplicación cliente estea baseada na web.

Ámbalas dúas propostas teñen as súas vantaxes e inconvenientes, que debemos analizar e valorar para chegar a definir a mellor solución posible. Se ben o rendemento e a robustez das aplicacións de escritorio son xa coñecidas, as ferramentas existentes actualmente nese ámbito para representación de información e creación de novos compoñentes visuais avanzados presentan carencias significativas e unha dificultade de uso e aprendizaxe elevada. Se pensamos por exemplo na construcción de interfaces gráficas mediante *Java Swing*, unha das ferramentas más amplamente utilizadas para este fin, non presenta demasiados problemas sempre que nos limitemos ó uso de compoñentes xa definidos ou á combinación dos mesmos, pero a dificultade aumenta drasticamente cando tratamos de construír compoñentes propios máis complexos.

Os avances que se levan producindo nos últimos anos no eido da web, atallan principalmente problemas deste tipo e tradúcense nunha mellora substancial da experiencia de usuario e da interactividade das aplicacións construídas, xunto con outros beneficios como mecanismos de actualización transparentes ó usuario, portabilidade, etc. Ademais, existen numerosas ferramentas e tecnoloxías neste ámbito que poderían simplificar de maneira importante a construcción de novos compoñentes visuais que satisfagan os requisitos expostos. Unha vantage importante que aporta o feito de definir unha ferramenta baseada na web é a ubicuidade, a posibilidade de que un asistente clínico poida realizar o seguimento dun determinado paciente dende calquera punto do mundo. Pola contra, o rendemento dunha aplicación deste tipo non está de todo claro, e podería non cumplir cas expectativas, dado o volume de datos que debemos manexar e a dificultade que suporía traballar con eles nun entorno coma ese.

Tendo todo isto en conta, e despois de realizar unha análise de ámbalas dúas propostas no contexto deste proxecto, decídese non adoptar unha postura conservadora, e apóstase pola creación dunha ferramenta baseada na web. O cliente apostá por un proxecto no que a innovación e a experimentación xoguen un papel importante, tentando aportar unha solución innovadora pero tamén eficiente, robusta e escalable, e tanto el como o equipo de desenvolvemento deciden afrontar os riscos que isto supón, asumindo que poderían chegar a provocar que o proxecto non rematase de forma satisfactoria, ben non cumprindo os requisitos ou ben rematando fóra do prazo dispoñible.

4.2. Arquitectura do sistema

As aplicacións RIA (Rich Internet Application) [11], nacen como unha combinación das vantaxes que ofrecen as aplicacións web e as aplicacións tradicionais, tentando mellorar a experiencia de usuario. A principal diferencia con estas é que non se producen recargas de páxina con cada chamada ó servidor, senón que estas se fan en segundo plano e non interfieren na visualización nin na interacción do usuario coa aplicación.

Un xeito simple de entender a súa arquitectura é pensar nunha arquitectura cliente-servidor [30] típica de aplicacións web, na que a lóxica de negocio, o control, e parte do modelo se trasladan ó cliente, como se amosa na figura 4.1. O cliente controla a interacción entre o usuario e a interface, xestiona a actualización de vistas e a presentación de datos, e controla tódalas peticionés de datos cara servidor. O servidor pola contra, ocúpase de procesar as peticionés do cliente, normalmente para gardar ou solicitar datos, e neste último caso, de darlle o formato apropiado para a súa comprensión por parte do cliente.

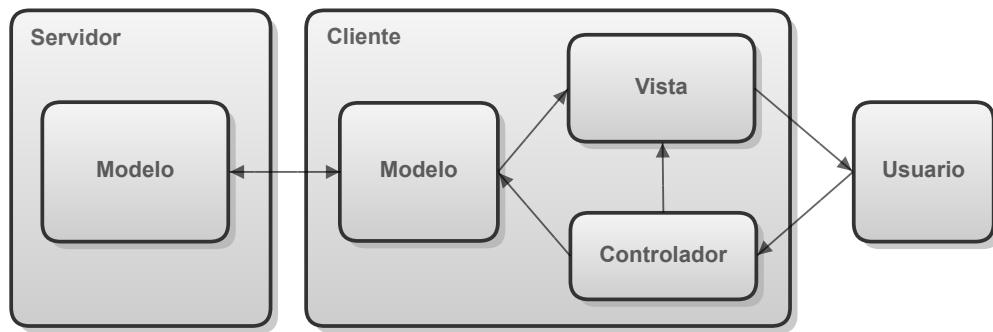


Figura 4.1: Arquitectura *RIA*

Pódese dicir, polo tanto, que o estado da aplicación reside no cliente, e non en sesións do servidor, como normalmente sucede na maioría de aplicacións web.

Visto isto, pasaremos a definir a arquitectura da nosa aplicación. Primeiro farémolo dende un punto de vista **físico** que nos permita comprender como estarán distribuídos os diferentes compoñentes hardware que conforman o sistema e o lugar que ocupan dentro de Servando, e seguidamente dende un punto de vista lóxico, que mostre a organización dos diferentes artefactos software, e as capas nas que se divide o sistema.

A nivel hardware a única componente do sistema é o sistema de información central, que se encargará de xestionar e almacenar os datos recibidos dende os dispositivos móbiles a través do “sistema xestor de dispositivos” de Servando, e ofrecerá mecanismos para o acceso e consulta dos mesmos. Este componente executarase nun servidor de aplicacións aloxado fisicamente no Complexo Hospitalario Universitario de Santiago de Compostela, e a súa ubicación dentro da arquitectura global da *Plataforma Servando* pode verse na figura 1.1, como xa se comentou anteriormente.

A nivel software existirán dúas capas ben diferenciadas, que á súa vez estarán divididas en módulos que se irán explicando en detalle neste e no seguinte capítulo:

- O *servidor*, que será responsable do acceso ó sistema de información, e de parte da lóxica de negocio. Ocuparase da xestión da información de seguimento recibida dende os dispositivos móbiles, e fará posible o acceso controlado á mesma.
- O *cliente* será responsable da presentación da información e da lóxica de interacción entre o usuario e o sistema. Esta capa manterá o “estado” da aplicación, e comunicarase co servidor únicamente para realizar a solicitude de información.

Na figura 4.2 amósase a totalidade do sistema, coas dúas capas lóxicas que se acaban de definir e os diferentes módulos ou servizos dos que consta cada unha delas. Como se pode ver, o cliente contén a maior parte da lóxica de negocio, limitándose a parte do servidor a permitir o acceso á información a través dos mecanismos apropiados que expón a capa de comunicacíons. A continuación explícanse en detalle as diferentes partes do sistema.

Comezando pola descripción **do cliente**, este terá un módulo principal ou **núcleo** composto por tres componentes básicas, que implementarán a funcionalidade necesaria para o funcionamento do sistema:

- O **xestor do escritorio** implementará as componentes típicas dun entorno de ventás, como poden ser unha barra de tarefas, un menú de aplicacións, soporte para lanzadores, etc. Permitirá ademais rexistrar tanto no escritorio coma no menú de aplicacións os componentes gráficos (normalmente ventás) dos diferentes módulos do sistema. Os detalles que nos levaron á elección deste tipo de interface serán comentados no capítulo 5.
- O **xestor de módulos** permitirá a carga dinámica de módulos, e activación e parada dos mesmos en tempo real. Deste xeito os recursos dun determinado módulo non será cargados ata que este se necesite, a non ser que o contrario se diga de xeito explícito, co que se cargarán durante o inicio da aplicación. Deste xeito diminuímos o uso de memoria e aumentamos o rendemento, non tendo que esperar pola carga de módulos que non van ser usados.
- Implementarase tamén un **xestor de notificacións** que ofrecerá mecanismos para mostrar avisos ó usuario de xeito rápido e sinxelo, e que os diferentes módulos implementados e os que se poidan implementar nun futuro utilizarán para realizar as súas notificacións de xeito unificado.

Sobre esta base implementaranse unha serie de subsistemas e módulos que permitirán satisfacer os diferentes requisitos de xestión e visualización existentes:

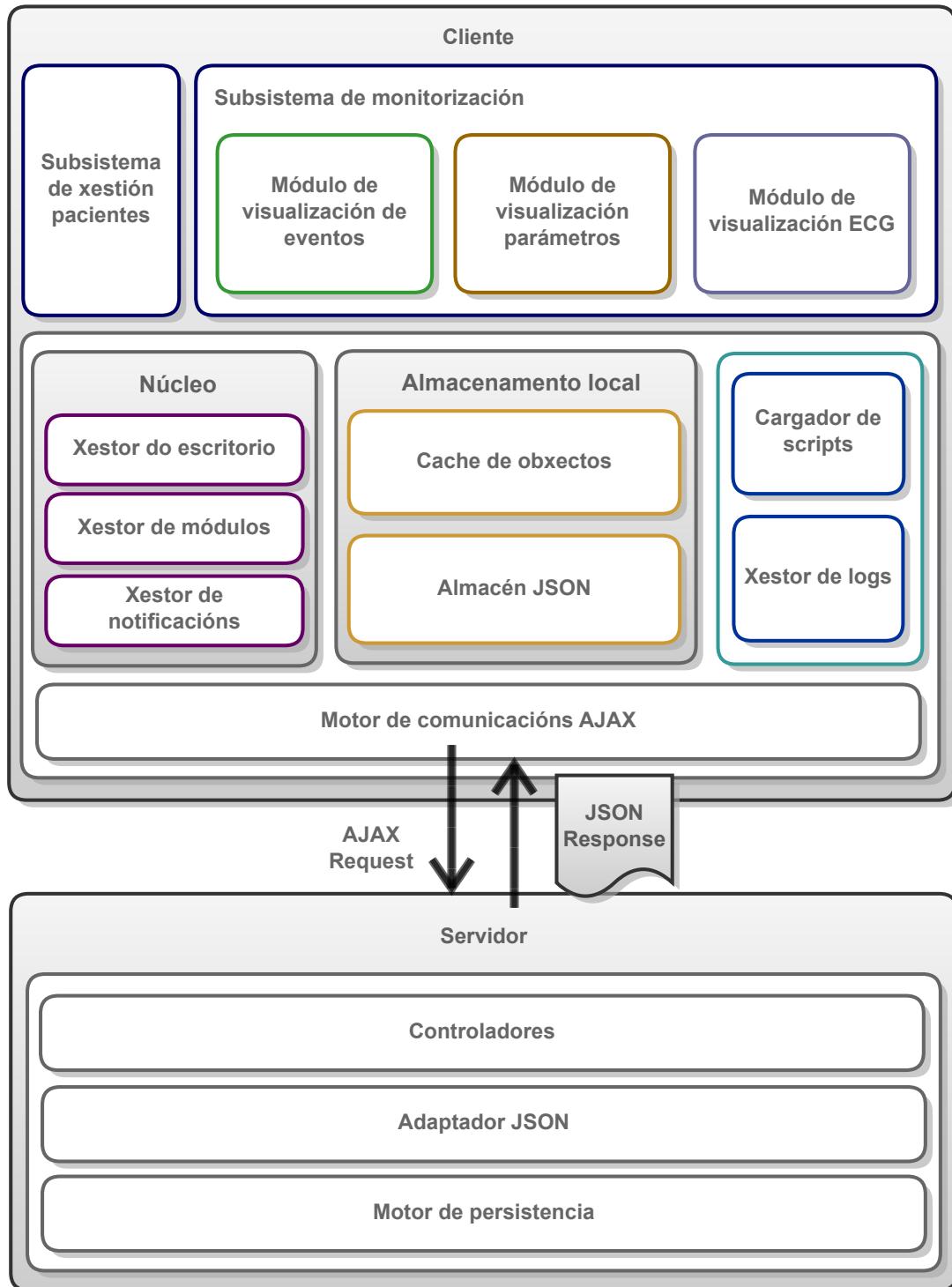


Figura 4.2: Diagrama de arquitectura do sistema

- O **subsistema de xestión de pacientes** será o encargado de implementar a funcionalidade que permita, como o seu nome indica, realizar a xestión de pacientes do sistema, isto é, as tarefas de creación, edición e borrado de pacientes e das súas correspondentes etapas de seguimento.
- O **subsistema de monitorización** será responsable da representación de toda a información do seguimento de pacientes. Estará dividido inicialmente en tres módulos independentes aínda que comunicados entre si, onde cada un deberá implementar os mecanismos de almacenamento e persistencia oportunos para o modelo de datos co que traballa. Os módulos que se implementarán serán os seguintes:
 - **Módulo de visualización de eventos:** Neste módulo implementarase a funcionalidade relativa á visualización de eventos, isto é, unha liña temporal interactiva na que se mostrarán cronoloxicamente os eventos dun paciente e o calendario que permitirá acceder ó seguimento realizado durante cada etapa.
 - **Módulo de visualización de parámetros:** Este módulo será o encargado de realizar as funcións de representación de parámetros, axuste de escalas, cuadrículas, etc. Dada a complexidade e volume dos datos que se deben representar, deberá ofrecer mecanismos de cacheado e procesado local dos mesmos.
 - **Módulo de visualización de ECG:** Neste módulo implementarase a funcionalidade referente á representación de electrocardiograma e de familias morfolóxicas, e os componentes necesarios para a solicitude de fragmentos de ECG de xeito remoto.
- O **cargador de scripts** ofrecerá a funcionalidade básica de cargar un ou máis ficheiros de xeito dinámico dende o servidor.
- O **xestor de logs** permitirá a xeración de informes que permitan ter constancia de tódolos eventos inesperados que poidan ocorrer durante a execución do sistema.
- Finalmente, para a comunicación co servidor, crearase un **módulo de comunicación AJAX**, que mediante esta tecnoloxía, explicada en detalle no apartado 4.4.6, permita a comunicación dos diferentes módulos co motor de persistencia para obter a información que precisen.

Con respecto á arquitectura proposta para o **servidor**, estará formada por tres componentes principais:

- Un conxunto de **controladores** que atenderá as peticións do cliente e realizará as tarefas oportunas para resolvelas. Será tamén o encargado de asegurar a autenticidade das diferentes peticións recibidas e de controlar o acceso a información.
- Un **motor de persistencia** que será o encargado de garantir a persistencia dos datos que precisa a plataforma Servando para o seu funcionamento e de proporcionar os mecanismos de acceso e tratamiento da mesma. Esta componente encargarase de traducir o modelo de obxectos da aplicación a un modelo relacional, e automatizará o acceso ó sistema xestor de bases de datos usado para almacenar a información.

- O cliente e o servidor comunicaranse mediante JSON, un formato de intercambio de datos que será detallado no apartado 4.4.5. Polo tanto crearase un conxunto de clases ás que nos referiremos como **Adaptador JSON**, e que se ocuparán transformar os datos obtidos da base de datos mediante o motor de persistencia a dito formato, para a súa posterior transferencia ó cliente.

4.3. Ferramentas de deseño

Nesta sección faremos unha descripción das ferramentas empregadas para o deseño dos diferentes compoñentes do sistema, que nos permitirán unha definición formal de cada un deles. Dado que modelaremos o sistema como un conxunto de obxectos que se relacionan entre si, escolleremos ferramentas que permitan abordar o deseño dende esta perspectiva e faremos uso de solucións probadas e amplamente utilizadas sempre que sexa necesario.

UML

UML (Unified Modeling Language) [14] ou Linguaxe Unificada de Modelado, é un conxunto de notacións gráficas, que permiten visualizar, especificar e documentar cada unha das partes que comprende o desenvolvemento de software.

UML utilízase para describir modelos do sistema, dos seus compoñentes e do seu comportamento, en diferentes niveis de detalle, e áinda que se poden chegar a describir os diferentes artefactos que componen a arquitectura do sistema e o seu comportamento a moi baixo nivel detalle ,non se trata dunha linguaxe de programación, pois simplemente mostra conceptos de deseño e non especifica un proceso ou metodoloxía a usar.

UML define 13 tipos de diagramas categorizados en tres grupos:

- **Diagramas de estrutura**, que se usan para describir aqueles elementos que deben estar presentes no modelo.
- **Diagramas de comportamento**, que mostran que debe suceder no sistema modelado. Dentro deste grupo atópanse os *diagramas de casos de uso* utilizados na sección 3.2.
- **Diagramas de interacción** que fan énfase no fluxo de control e de datos entre os elementos do sistema.

No noso caso únicamente se usarán dous dos diagramas do conxunto disponible, os diagramas de clases e os diagramas de secuencia.

Os diagramas de clases son un tipo de diagrama estático que permite modelar solucións de deseño e arquitectura, mostrando conxuntos de clases, atributos e relacións de compoñentes do sistema ou da súa totalidade. Este tipo de diagramas permitirános definir a estrutura conceptual da información que manexará o sistema, os artefactos que se encargarán do seu funcionamento, e as relacións entre uns e outros.

Os diagramas de secuencia, pola contra, mostran a interacción dun conxunto de obxectos da aplicación, e permiten o modelado de escenarios máis concretos da execución do sistema, incluíndo os obxectos e clases que interveñen e os mensaxes intercambiados. Durante o deseño, usaremos este tipo de diagramas para modelar a interacción entre os diferentes compoñentes de cada módulo do sistema ou cando sexa necesario representar o intercambio de mensaxes ou o fluxo de control entre varios artefactos.

Patróns de deseño

Os patróns de deseño son a base para a procura de solucións xenéricas a problemas comúns no desenvolvemento de software, ou o que é o mesmo, brindan unha solución efectiva, probada e documentada a problemas de desenvolvemento de software en contextos similares.

Habitualmente, a especificación dun patrón de deseño consiste na definición dunha serie de clases relacionadas entre si, e dunha serie de responsabilidades que cada unha delas asume dentro dese conxunto para resolver o problema concreto para o que se propuxo. Son polo tanto expresados normalmente como simples diagramas de clases.

Os patróns de deseño clasifícanse en función do tipo de problema ó que tratan de poñer solución, en tres grupos:

- **Patróns creacionais:** Refírense ós patróns que tratan os diferentes mecanismos de creación de obxectos, tratando de realizar esta tarefa da maneira máis axeitada á situación que describe o problema.
- **Patróns estruturais:** Facilitan o deseño identificando a forma máis simple e útil de componer e relacionar os obxectos.
- **Patróns de comportamento:** Identifican patróns comúns de comunicación entre os diferentes compoñentes dun sistema, proporcionando un esquema flexible para a interacción e distribución de responsabilidades entre os obxectos.

A utilización deste tipo de solucións pode facilitarnos o deseño do sistema sempre que o fagamos na súa xusta medida, pois, a súa utilización de xeito abusivo pode incrementar innecesariamente a complexidade do sistema e mermar de xeito significativo o rendemento do mesmo.

A continuación móstranse os patróns que foron de utilidade para o desenvolvemento do sistema, e explícase a utilidade que estes teñen para a elaboración de aspectos concretos do sistema.

Modelo-Vista-Controlador (MVC) [15]

Categoría: Patrón estrutural

O patrón MVC ten como obxectivo separar os datos dunha aplicación, a súa presentación e a lóxica de control en tres compoñentes separados:

- O *Modelo* encárgase da representación específica da información ca que opera o sistema.
- A *Vista* presenta a información nun formato adecuado que permita a interacción co usuario a través, normalmente, dunha interface gráfica.
- O *controlador* ocúpase de xestionar fluxo de eventos entre a vista e o modelo, permitindo o seu desacoplamento.

Neste proxecto, farase un dobre uso deste patrón:

Por unha parte, utilizarase para separar o cliente do servidor, de xeito que o servidor se ocupe do manexo da información, o cliente da súa presentación e o controlador sirva de nexo entre ámbolos dous e xestione a súa interacción.

Por outra banda, será usado no cliente para separar os diferentes compoñentes gráficos da lóxica de presentación e do acceso ó sistema de almacenamento local.

Fachada (*Facade*) [15]

Categoría: Patrón estrutural

O patrón *Fachada* permite ofrecer unha interface simple para un subsistema complexo, ou establecer puntos de acceso unificados para un conxunto máis amplio de compoñentes.

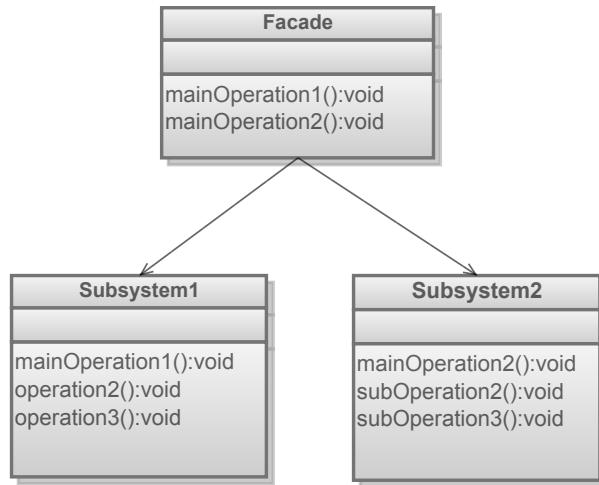


Figura 4.3: Diagrama de clases do patrón *Facade*

Este patrón será utilizado para ofrecer unha interface unificada á lóxica do servidor, permitindo o desenvolvemento independente desta parte do sistema e do cliente. Ademais, permitirá proporcionar interfaces para os diferentes módulos e ocultar deste xeito a complexidade interna de cada un deles.

Instancia única (*Singleton*) [15]

Categoría: Patrón creacional.

Este patrón úsase cando se deseja restrinxir o uso dunha determinada clase á creación dunha única instancia.

Normalmente, en proxectos coma este, existen numerosos componentes da interface que non ten sentido duplicar. Para resolver problemas deste tipo, faremos uso desta solución.

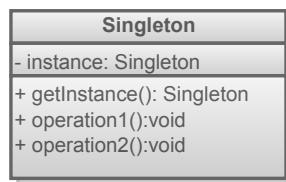


Figura 4.4: Diagrama de clases do patrón *Singleton*

Vista composta (*Composite view*) [15]

Categoría: Patrón estrutural

O patrón *vista composta* permite a construción de obxectos complexos a partir de outros máis simples e similares entre si, realizando unha composición recursiva en estrutura de árbore. Dado que todos eles teñen unha interface común, poden ser tratados da mesma maneira simplificando así o seu manexo.

No diagrama 4.5 poden verse os participantes neste patrón e as relacións entre eles.

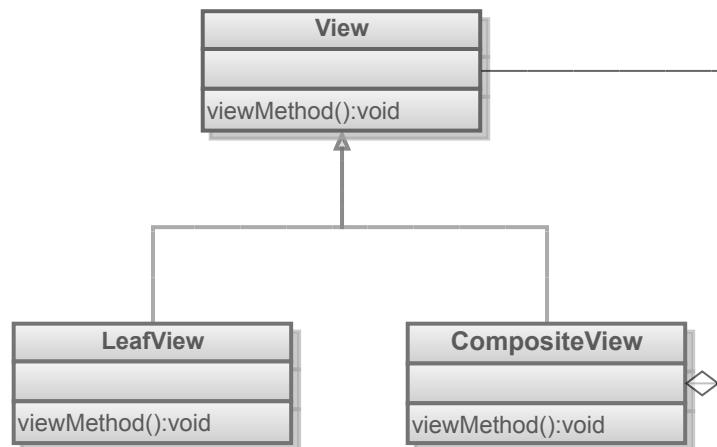


Figura 4.5: Diagrama de clases do patrón *Composite View*

Axudante da vista (*View Helper*) [15]

Categoría: Patrón de comportamento

O patrón *Axudante da vista* permite que unha determinada clase, normalmente unha vista (aínda que tamén pode unha clase non relacionada con tarefas deste tipo), se apoie noutras (*helpers*) para a realización dunha determinada tarefa.

No diagrama 4.6 poden verse os participantes neste patrón e as relacións entre eles.

Neste proxecto este patrón será útil para, entre outras cousas, a creación de clases de apoio á interface, na parte do cliente, e para a implementación do conxunto de clases que realizan as función de tradución ao formato JSON, na parte do servidor.



Figura 4.6: Diagrama de clases do patrón *View Helper*

Observador (*Observer*) [15]

Categoría: Patrón de comportamento.

Este patrón ofrece un modelo de notificación de eventos entre obxectos que permite minimizar o acoplamento entre eles.

Como se pode ver no diagrama da figura 4.7, o observador (*Observer*) proporciona os métodos *attach()* e *detach()* que permiten que os demás suxeitos (*Observable*) se rexistren, e estes a súa vez implementan o método (*update()*) que permite que o observador lles notifique os eventos.

Este patrón será amplamente utilizado para a construcción de componentes da interface de usuario da parte do cliente, permitindo o desacoplamento da visualización e da lóxica de control. O patrón MVC comentado no apartado 4.3 será usado conxuntamente con este para lograr tal resultado.

4.4. Ferramentas de desenvolvemento

A continuación exponse o conxunto de ferramentas e tecnoloxías que serán usadas para o desenvolvemento. Primeiro farase unha breve introducción da función que cada unha delas realiza dentro do sistema, utilizando a figura 4.8, e posteriormente comentaranse más polo miúdo de xeito individual.

Como se ve en dita figura, a parte do cliente estará construída sobre Javascript e nela utilizaranse principalmente dúas ferramentas. Por unha banda utilizarase o *framework Ext JS* para

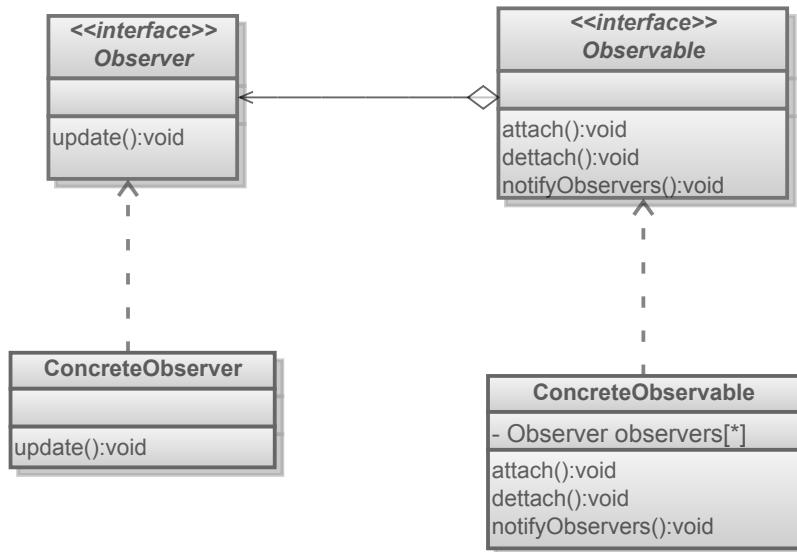


Figura 4.7: Diagrama de clases do patrón *Observer*

a construcción dos diferentes compoñentes da interface de usuario, e por outra parte a libraría *Highcharts* para o desenvolvemento dos gráficos necesarios para a presentación de datos. A parte do servidor estará baseada en Java, e sobre esta linguaxe utilizaranse ferramentas como a API de Servlets para proporcionar o acceso remoto, *JSONObject* para a tradución dos obxectos Java ó formato de transporte, e *Hibernate* como motor de persistencia.

4.4.1. Java SE

Como xa se comentou no capítulo anterior, unha das restriccións para a construcción dos módulo de persistencia é que estea desenvolvido en tecnoloxía Java (requisito RD.1), permitindo así a comunicación directa co “Sistema xestor de dispositivos” de Servando. Neste contexto, a plataforma Java resulta particularmente axeitada, ofrecendo unha gran cantidade de tecnoloxías e servizos para a computación distribuída.

Para o desenvolvemento en Java empregaremos o IDE Netbeans. Os motivos que nos levaron a seleccionar este producto é o seu carácter libre, multiplataforma, e a ampla experiencia no seu uso coa que contamos.

4.4.2. Hibernate

A automatización dos procesos de almacenamento e recuperación de datos en sistemas complexos reporta unha serie de vantaxes coma a redución do tempo de desenvolvemento, independencia do xestor de bases de datos, escalabilidade, flexibilidade, e a facilidade de xestión e mantemento dun sistema orientado a obxectos.

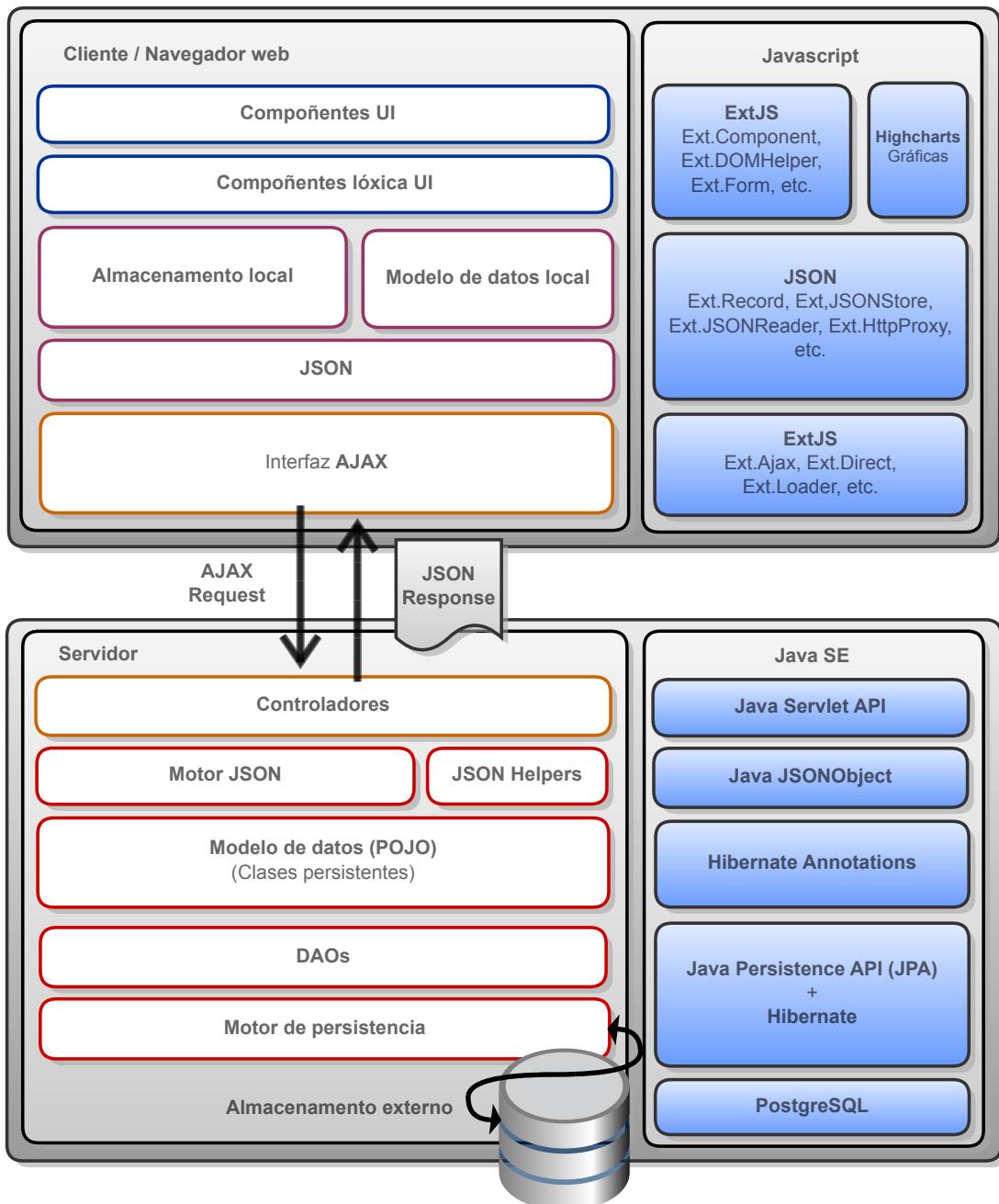


Figura 4.8: Ferramentas e tecnoloxías empregadas para o a construción das diferentes partes do sistema

Actualmente existen varias alternativas á hora de escoller unha ferramenta de mapeo obxecto-relacional. A maioría delas céntranse en dar unha implementación da *Java Persistence API (JPA)* incluída no estándar EJB3. Entre as distintas opcións, as más populares, e que polo tanto será necesario analizar para proceder a elección daquela que se adapte mellor ós requisitos do presente

proxecto, atópanse *OpenJPA* [10], *Hibernate* [19], *EclipseLink* [12] e *TopLink* [24].

Os criterios que se usarán para elixir unha das alternativas son:

- **Rendemento:** É o factor máis importante a ter en conta, pois condiciona directamente a eficiencia da aplicación.
- **Escalabilidade:** É tamén un factor importante, xa que hai unha alta probabilidade de ter que ampliar o modelo de datos coa incorporación de novos servizos a Servando.
- **Documentación:** Dada a escasa experiencia previa con ferramentas deste tipo, será importante contar cunha boa documentación.

Dado que a realización dunha análise deste tipo implicaría un coñecemento mínimo de cada unha das alternativas, e un investimento de tempo considerable que dadas as características do proxecto non é posible realizar, procederemos á utilización de comparativas publicadas para decantarnos por unha opción concreta.

A primeira das comparativas analizadas será o *JPA Benchmark* [6]. Nesta comparativa avalíanse as ferramentas *Hibernate*, *EclipseLink* e *OpenJPA*. Nela realizanse 6 tests diferentes nos que se proban operacións de inserción, consulta, actualización e borrado con diferentes tamaños de transacción. A táboa 4.1 amosa os resultados obtidos por cada unha das alternativas nos diferentes tests realizados.

	Hibernate	EclipseLink	OpenJPA
Entidades básicas	8.5	8.3	0.15
Coleccións	7.0	2.6	failed
Herencia	7.1	7.2	0.15
Indexado	4.7	-	0.23
Modelo árbore	0.99	1.1	failed
Multithreading	16.8	15.9	failed
Media	7.7	7.3	0.18

Táboa 4.1: Resultados da comparativa de ferramentas de mapeado obxecto-relacional de [6]

Pódese ver como, de media, *Hibernate* está un pouco por enriba de *EclipseLink*, aínda que este o supera en algunas das probas. Os resultados obtidos por *OpenJPA* son significativamente inferiores, chegando a non superar algúns dos tests realizados.

A segunda comparativa que usaremos [4], inclúe as catro alternativas expostas previamente. Nesta comparativa faise un análisis do consumo de memoria de cada unha delas.

Os resultados en canto a consumo de memoria móstranse na táboa 4.2 e favorecen a *TopLink* e *Hibernate* case por igual, metras que *EclipseLink* e *OpenJPA* saen desfavorecidos.

En canto á escalabilidade das diferentes ferramentas, todas elas se basean en **JPA**, polo que presentan características similares. Onde si podemos atopar diferenzas é na documentación, sendo

	Hibernate	EclipseLink	OpenJPA	TopLink
Máximo de memoria utilizado	25 mb	60 mb	57 mb	25 mb
Memoria ocupada ó final da proba	13 mb	43 mb	32 mb	15 mb

Táboa 4.2: Resultados da comparativa de ferramentas de mapeado obxecto-relacional de [4]

as de *Hibernate* e *OpenJPA* as más completas e mellor estruturadas, seguidas por *TopLink* e *EclipseLink*.

Tendo en conta os puntos anteriores, a opción seleccionada é *Hibernate*, posto que respondeu sensiblemente mellor có resto de alternativas nas probas realizadas, e ademais é unha ferramenta de código aberto e totalmente gratuítia.

Hibernate é unha ferramenta que permite realizar o mapeado entre o modelo de obxectos dunha aplicación e o modelo entidade-relación das bases de datos en entornos Java. O término utilizado normalmente é *ORM (Object/relational mapping)* e consiste en realizar a transición dunha representación dos datos nun modelo relacional a un modelo orientado a obxectos e viceversa. Deste xeito, cando deseñamos o modelo de datos establecemos as regras que se usarán para facer a tradución, e non deseñamos a base de datos directamente.

Hibernate está deseñado para ser flexible en canto ó esquema de táboas utilizado, e ademais de realizar esta transformación do modelo, proporciona capacidades para a obtención e almacenamento dos datos [22].

Para establecer as regras de transformación do modelo de datos usaremos o compoñente *Hibernate Annotations* [18], que é parte desta plataforma e implementa o estándar *Java Persistence Api (JPA)* [21]. O uso deste compoñente baséase na introdución de anotacións sobre cada clase Java, especificando a maneira na que se desea persistir e como van ser traducidos os seus atributos. Estas anotacións son posteriormente procesadas por *Hibernate* para realizar o mapeado de xeito dinámico.

4.4.3. PostgreSQL

Como sistema de xestión de bases de datos usarase *PostgreSQL*, unha ferramenta *open source* que permite a xestión de bases de datos relacionais.

Durante a elección de feramentas barallouse a posibilidade de usar *MySQL*, pero finalmente seleccionouse *PostgreSQL* debido a un bug [1, 2] do conector de MySQL para Java (*Connector/J*) no manexo de campos binarios (*blobs*) de gran tamaño que obriga a cargalos completamente en memoria cando se quere acceder a un anaco do seu contido, e que perxudicaría notablemente o rendemento do sistema, dado que este tipo de estruturas será utilizado para almacenar os valores dos diferentes sinais, como se detalla no apartado 5.4.1.

4.4.4. Javascript

Javascript é unha linguaxe de programación interpretada, orientada a obxectos, e baseada en prototipos. En Javascript non existen clases como tal, e a orientación a obxectos conséguese a través da clonación de obxectos xa existentes, que serven de prototipos, estendendo as súas funcionalidades. Este estilo de programación coñécese tamén como *programación baseada en instancias*.

Esta linguaxe utilízase principalmente, na programación de aplicacións web no lado cliente, executándose nun navegador, e permitindo a mellora da interface de usuario e a creación de páxinas web dinámicas.

Neste proxecto, será a linguaxe base que usaremos para implementar a aplicación cliente.

4.4.5. JSON

JSON é un formato lixeiro de intercambio de datos, baseado nun subconjunto do da linguaxe de programación Javascript, simple de ler e escribir para humanos, e fácil de xerar e interpretar para máquinas. Consta de dúas estruturas básicas:

- Unha colección de pares nome/valor, coñecida en moitas linguaxes como *obxecto*, rexistro, estrutura, ou vector asociativo.
- Unha lista ordenada ou vector de valores.

Estas estruturas son universais e están virtualmente soportadas dunha forma ou outra en tódalas linguaxes de programación, logrando así a independencia requirida para un bo formato de intercambio de datos.

Neste proxecto, será o formato no que se comunicarán o cliente e o servidor, e no que intercambiarán toda a información relativa ós datos de seguimento de pacientes.

4.4.6. Ajax

Ajax (*Asynchronous JavaScript And XML*), é unha ferramenta de desenvolvemento web, que axuda á creación de aplicacións RIA (*Rich Internet Applications*).

Realmente non é unha tecnoloxía en si mesmo, senón que é un conxunto de tecnoloxías que se unen mediante JavaScript, para permitir manter unha comunicación asíncrona co servidor en segundo plano, logrando que a solicitude e transferencia de datos non interfiran coa visualización nin co comportamento da páxina:

- *Document Object Model (DOM)* [32] permite interactuar de xeito programático cos elementos da páxina.
- *XMLHttpRequest* [34] posibilita o intercambio asíncrono de información entre a capa do cliente e a capa do servidor.

- *XML* [33] ou *JSON* son usados maiormente como formato de transmisión de datos, ainda que se podría usar calquera outro formato de texto plano.
- Como xa dixemos, Javascript une o resto de compoñentes, e permite a utilización das mesmas.

Ajax é unha técnica válida para múltiples sistemas operativos e plataformas, pois está baseado en estándares abertos coma os citados Javascript ou *Document Object Model (DOM)*.

No presente proxecto, usarase baixo o soporte de Javascript e xunto con *JSON*, para realizar a comunicación entre o cliente e o servidor.

4.4.7. Librarías externas

JSONObject

Dado que na parte do servidor traballaremos en Java con obxectos, e o para comunicarnos co cliente deberemos empregar o formato *JSON* comentado anteriormente, precisamos dalgunha utilidade ou ferramenta que nos permita facer unha tradución entre ámbolos dous modelos. Neste caso, usaremos a libraría *JSONObject*.

JSONObject é unha libraría de carácter libre e gratuíto baseada en Java, que permite a xeración directa de estruturas en formato *JSON* dende obxectos Java, ou ben a construcción dunha estrutura deste tipo de xeito programático a partires dun ou máis obxectos.

Highcharts

Para a construcción do cliente, deberemos facer uso de ferramentas ou librarías que nos permitan a xeración de gráficas interactivas. Aínda que hai multitud de ferramentas dispoñibles con estas características (*Google Chart Tools* [16] ou *dygraphs* [5] entre outras), a elixida neste caso foi *Highcharts* [29].

Highcharts é unha libraría de uso gratuíto para aplicáis non comerciais, escrita totalmente en Javascript, e que permite a creación de gráficos interactivos de diversos tipos. As razóns que nos levaron a optar por esta libraría, a parte do seu carácter gratuíto, son as seguintes:

- Acadou un alto rendemento e presentou maior fluidez ca o resto de alternativas [16, 5] en probas realizadas cun alto volume de datos, chegando sen problemas ata os 3000 datos por serie, mentres que o resto comezaban a perder rendemento arredor dos 1000.
- As gráficas xeradas gozan dunha vistosidade excelente.
- É de código aberto, o que nos permitirá estender a súa funcionalidade segundo precisemos, e expón un conxunto de funcións, eventos, e propiedades que nos permiten engadir novas funcións sen ter que modificar o seu código.
- Por último, e destacando notablemente por enriba do resto de alternativas, dispón dunha moi boa documentación.

Ext JS

Ext JS [27] é unha libaría que permite a construción de aplicacíons Web avanzadas ou RIA (*Rich Internet Applications*). Destaca como un dos mellores *frameworks* para o desenvolvemento en Javascript, proporcionando un amplo conxunto de compoñentes gráficos de alto nivel, un intuitivo e extensible modelo de compoñentes, e unha *API* de fácil uso permitindo crear aplicacíons web robustas e multiplataforma. Do mesmo xeito, facilita un conxunto de clases que permiten a implementación de comunicacíons *Ajax*, prové dun manexo de *layouts* similar ao de *Java Swing*, e proporciona un estilo visual atractivo e consistente entre diferentes plataformas, que pode ser modificado facilmente.

Actualmente *Ext JS* está sendo usado por un gran número de compañías de renome como *Adobe*, *Amazon*, *Panasonic*, *Pixar*, *Siemens* ou *Sony*.

Neste proxecto será usado como *framework* de desenvolvemento base para implementar os diferentes compoñentes do cliente, utilizando os mecanismos de creación de obxectos, extensión, manexo de eventos, e manipulación do DOM que ofrece.

4.5. Ferramentas de validación

A validación da ferramenta será explicada en detalle no capítulo 6. Porén nesta sección comentaranse as ferramentas que se utilizarán para a validación do sistema. Para a validación dos compoñentes do servidor farase uso de JUnit, mentres que para os da interface de usuario se utilizará a avaliación heurística.

4.5.1. JUnit

JUnit é unha ferramenta que permite a realización de probas unitarias en entornos Java. A súa utilización é sinxela, pois baséase na avaliación, para cada un dos métodos que se deben probar, do resultado esperado da súa execución e do resultado obtido durante a proba para unha entradas concretas. Se o resultado cumpre coa especificación, JUnit devolverá que o método pasou a proba satisfactoriamente, mentres que se o resultado é diferente do esperado indicará que houbo un fallo no método correspondente.

Este xeito de realizar as probas permite comprobar que o código cumpre cos requisitos anteriores cando se realizan cambios sobre el ou se fai algúin tipo de refactorización, asegurando que a funcionalidade non se viu alterada.

4.5.2. Avaliación heurística

A avaliación heurística baséase na análise dos diferentes elementos da interface de usuario para determinar se estes cumplen cuns principios de usabilidade establecidos. Debemos pois, previamente, aclarar o concepto de usabilidade.

A usabilidade pódese describir, de xeito informal, como a facilidade ca que as persoas poden interactuar cun determinado sistema. Dun xeito máis formal, podemos remitirnos a múltiples definicións:

- Segundo a norma ISO/IEC FDIS 9126-1, Software Engineering-Product Quality-Part [8], a usabilidade defíñese como a capacidade dun producto para ser entendido, aprendido, e usado, ademais de resultar atractivo para o usuario, cando este é usado baixo unhas condicións específicas.
- Segundo a norma ISO 9241-11, Guidance on Usability [7], defíñese a usabilidade como a medida na que un producto pode ser usado por determinados usuarios para conseguir obxectivos específicos con efectividade, eficiencia e nun contexto de uso específico.

A avaliación heurística analiza a usabilidade dunha interface atendendo a unha serie de principios recoñecidos de usabilidade. Unha avaliación heurística pode ser tan flexible ou rigorosa como se desexe, pero existen numerosas listas de principios heurísticos a avaliar en calquera sistema, como por exemplo “as 8 regras de ouro de Schneidman” [28] ou “as 10 regras heurísticas de usabilidade de Nielsen” [23] que garanten a detección da maioría de problemas deste tipo.

Segundo este último, os aspectos que se deben ser avaliados e tidos en conta para garantir a usabilidade do sistema son os seguintes:

- *Visibilidade do estado do sistema:* O sistema debe manter ós usuarios informados do seu estado, mantendo proporcionando a retroalimentación apropiada.
- *Utilizar a linguaxe dos usuarios:* A linguaxe do sistema (frases, termos, conceptos, etc.) deben ser adecuados á natureza dos usuarios que van a usalo.
- *Control e libertade:* O usuario débese sentir libre para navegar pola aplicación, e ter a posibilidade de saír facilmente de estados nos que entrou por erro.
- *Consistencia e estándares:* O sistema debe seguir as normas e convencións da plataforma sobre a que está implementado, mantendo a consistencia interna entre os seus diferentes componentes.
- *Prevención de errores:* Débese previr a aparición de errores, no canto de xerar boas mensaxes de erro.
- *Minimizar a carga da memoria do usuario:* Débese potenciar unha interacción guiada e simple que non obrigue ó usuario a memorizar o camiño dunha determinada acción. As opcións más importantes deben atraer a atención do usuario e permanecer visibles sobre aquelas que non teñen tanta importancia.
- *Flexibilidade e eficiencia de uso:* As ferramentas do sistema deben ser autoexplicativas, a axuda debe estar visible, e débense proporcionar accesos directos para aquellas funcións que sexa posible.

- *Diálogos estéticos e deseño minimalista:* Toda a información exposta debe ser útil e ter unha función, non debendo mostrarse aquela que raramente se utiliza e pode distraer o usuario e interferir na percepción das opcións realmente importantes.
- *Axudar ó usuario a recoñecer e recuperarse dos errores:* En caso de ter que mostrar mensaxes de erro, estas deben indicar de forma precisa o problema, e suxerir solucións aos problemas sempre que sexa posible.
- *Axuda e documentación:* A axuda debe ser simple, non demasiado extensa, e estar enfocada ás tarefas do usuario.

Capítulo 5

Deseño e implementación

Neste capítulo afondarase nos detalles de deseño e implementación da ferramenta. En primeiro lugar analizarase o modelo de datos co que traballará a aplicación e a arquitectura de información proposta para solucionar os diferentes problemas que acarrexa a xestión da gran cantidade e diversidade de información que se xera durante o seguimento e a súa presentación ó usuario. A continuación presentarase a interface gráfica da aplicación, describindo os diferentes compoñentes que a conforman, e por último comentarase desde unha perspectiva técnica o deseño e implementación dos diferentes artefactos que componen o servidor e o cliente.

5.1. Modelo de datos

Nesta sección describiremos o modelo de datos co que traballaremos, facendo fincapé no modelado daquelas estruturas de información máis complexas, que comportan unha maior dificultade á hora de traballar con elas. En xeral, toda a información obtida a través dos dispositivos de seguimento de Servando ten carácter temporal, pois reflicte unha serie de acontecementos que se producen durante a monitorización dun determinado paciente, con maior ou menor grao de abstracción. Esta idea reflíctese no modelo de datos deseñado. As clases que o representan e as relacións entre elas poden verse na figura 5.1.

Como xa comentamos no apartado 3.1 no nivel máis alto sitúase o evento, e a súa propiedade básica é o instante no que ten lugar. A interface *IEvent* obriga ás clases que a implementan a proporcionar o método *getTimestamp()* que permite obter o instante no que se produciu un determinado evento. Desta interface derivan outras dúas. Por unha banda, a interface *ISystemEvent* da que penden tódolos eventos relativos ó funcionamento do sistema (*SensorDisconnection*, *SensorReconnection*, *BatteryLevel* e *BluetoothConnectivityError*.), e por outra a interface *IMedicalEvent* da que implementan o resto de eventos de carácter médico.

Decidiuse agrupar a información obtida de cada paciente (clase *Patient*) en etapas de seguimento (clase *MonitoringEpisode*) que permitisen simplificar a xestión dun volume tan elevado de datos, e a monitorización dun mesmo paciente en diferentes etapas temporais. A interface *IMedicalEvent*

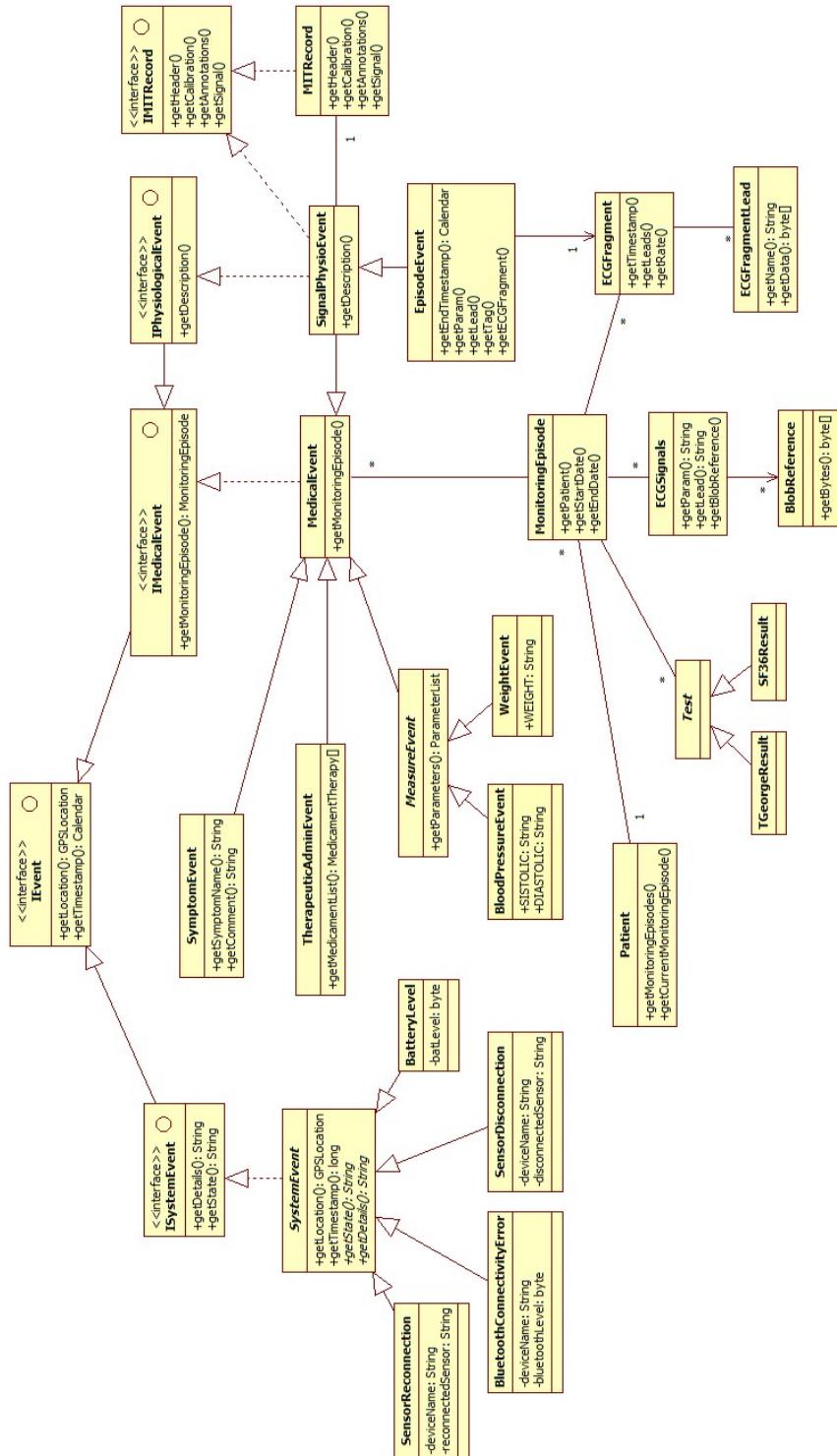


Figura 5.1: Modelo de datos

obriga a proporcionar o método *getMonitoringEpisode()*, que permite obter a etapa á que pertence un determinado evento, e polo tanto o paciente ó que este fai referencia.

A clase *MedicalEvent* implementa a interface *IMedicalEvent* para evitar a implementación en diversas clases dos métodos comúns ós diferentes eventos como a obtención da etapa de seguimento, do paciente ou da súa data, e dela derivan unha serie de clases que modelan o conxunto de eventos que se poden producir no sistema:

- A clase *SymptomEvent* modela os eventos de diversos síntomas que poida comunicar o paciente (mareos, dor de cabeza, etc.). Este tipo de eventos caracterízase por un *nome de síntoma* e un comentario acerca do mesmo.
- A clase *TherapeuticAdminEvent* modela os eventos de administración terapéutica, que consisten nun conxunto de terapias aplicadas ó paciente. Por exemplo, inxerir un comprimido dun determinado medicamento.
- A clase *MeasureEvent* modela diferentes medicións puntuais sobre algúns parámetros fisiológicos (peso, tensión arterial) que se lle realizan ao paciente ó longo do tempo. Esta clase é abstracta, e declara o método *getParameters()*, que cada subclase deberá implementar en función dos parámetros que poidan ser medidos. A clase *BloodPressureEvent* esténdea para o caso da medición da tensión arterial e a clase *WeightEvent* para a medición do peso.
- A clase *SignalPhysioEvent* permite modelar eventos que poidan ir acompañados dun fragmento de sinal especificado no formato de almacenamiento de datos MIT-BIH. Este formato comentase en detalle en [17], e a clase *MITRecord* permite a súa utilización.
- Os episodios (clase *EpisodeEvent*), modélanse como eventos, engadíndolle unha data de finalización, e pasando así a ter unha duración determinada. As clases *ECGFragment* e *ECGFragmentLead* permiten que un episodio referencia fragmentos de electrocardiograma de ata 5 minutos, que poidan ser enviados conxuntamente co resto de información á que fai referencia.

Por outra banda, unha etapa de seguimento deberá conter toda a información referente ós sinais de parámetros de electrocardiograma obtidos durante a monitorización. Este tipo de datos, rexistrado cunha alta frecuencia, pode chegar a recibirse de xeito continuado durante un longo intervalo de tempo, producindo un alto volume de información. Para manexala creouse a clase *ECGSignals*, que permite gardar os diferentes valores dun parámetro concreto en forma de array de bytes (*byte[]*), quedando estes compactados.

5.2. Arquitectura da información

A arquitectura da información é un proceso transversal ó deseño dunha interface, que ten como obxectivo que a asimilación dos contidos por parte do usuario sexa eficiente e efectiva.

Como se pode ver no modelo de datos, a maior parte da información coa que teremos que traballar ten carácter temporal. Porén non toda a información se obtén coa mesma frecuencia,

chegando a variar esta dende semanas ou días ata fraccións de segundo, e xerando un gran volume de información.

Para xestionar esta enorme cantidade de datos, e permitir que o usuario a visualice e analice no entorno no que o vai ter que facer (xa de por sí sobrecargado de información) é necesario ter en conta os seguintes puntos:

- **Procesar e dosificar** a enorme cantidade información que se manexa, mostrándoa de xeito claro para que o usuario a comprenda, e priorizando aquela que é máis relevante.
- **Organizar, estruturar e distribuír** esa información de xeito correcto, co fin de que o usuario poida facer da experiencia de recuperación algo simple, agradable, eficaz e produtivo.
- **Integrar interactividade, navegación e contido** co fin de que a información se traspase do sistema ó usuario sen dificultade.

Para conseguir unha asimilación da información eficiente e agradable por parte do usuario, e guialo durante a navegación proponse a estruturación da información en distintos niveis de abstracción, tendo en conta a súa granularidade temporal, e reflexando esta división na disposición dos diferentes compoñentes na pantalla. Como se ve na figura 5.2, a medida que descendemos na pantalla o nivel de detalle aumenta, creando unha especie de pirámide invertida na que a información se organiza xerarquicamente.

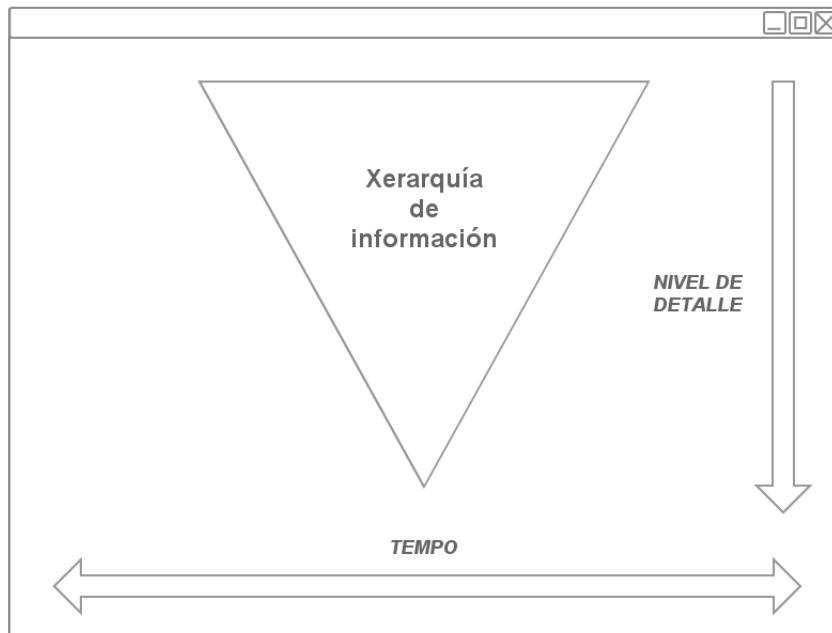


Figura 5.2: Arquitectura da información

Nos niveis superiores aparecerá información relativa a eventos concretos e relevantes para o seguimento, na que o usuario debería centrarse en primeiro lugar. Se o usuario precisa afondar nos detalles do suceso, descende polos seguintes niveis de información, analizándoo cun menor grao de

abstracción. Se pola contra a información que se mostra no primeiro nivel é suficiente para que o usuario comprenda o suceso, non será preciso que faga unha inversión de tempo maior tratando de caracterizar o acontecemento, o que é realmente importante no entorno de sobrecarga de información no que o persoal clínico debe desenvolver as súas funcións.

Por outra banda, o usuario visualiza en cada nivel de información unha ventá temporal concreta, sendo sempre a ventá dos niveis inferiores unha fracción temporal da dos niveis superiores, e pódese desprazar temporalmente adiante e atrás simultaneamente en todas elas.

5.3. Deseño da interface de usuario

A interface e o compoñente que se encarga de establecer a comunicación entre o sistema e o usuario, polo que resulta unha das partes fundamentais de calquera aplicación. Nesta sección expoñeremos a estrutura global da interface proposta e a continuación detallaremos cada un dos seus compoñentes dende unha perspectiva de deseño, tanto gráfico como de interacción.

5.3.1. Estrutura da interface

Como se comentou na sección 4.1, é moi probable que a funcionalidade e os servizos ofrecidos por Servando se vexan incrementados nun futuro. Os novos servizos non terán necesariamente que estar enfocados o seguimento, (recordemos o suposto servizo de citas do que falamos no capítulo anterior), polo que é necesario que a interface deseñada permita a inclusión de novos compoñentes independentes, enfocados a tarefas distintas, e todo elo baixo un mesmo entorno que permita a interacción co seu conxunto de xeito cómodo e eficaz.

Para conseguir isto, proponse unha interface similar á dun escritorio tradicional, na que exista un menú principal que recolla os diferentes servizos dispoñibles, e un entorno de ventás que permita facer uso de calquera deles utilizando todo o espazo dispoñible. As figuras 5.3 e 5.4 amosan o deseño da interface realizada. A primeira delas mostra o escritorio baleiro unha vez se iniciou o sistema, mentres que a segunda mostra o conxunto de compoñentes implementados.

5.3.2. Compoñentes da interface

A continuación comentaremos o conxunto de compoñentes que conforman a interface de usuario. Comezaremos comentando a ventá de administración de pacientes, dende a que se pode acceder ó seguimento. Despois comentaremos a ventá de seguimento e as compoñentes que a forman, detallando o deseño e obxectivo de cada unha delas, dende as que se encargan de representar a información de máis alto nivel ata aquelas que representan a información de carácter máis específico.

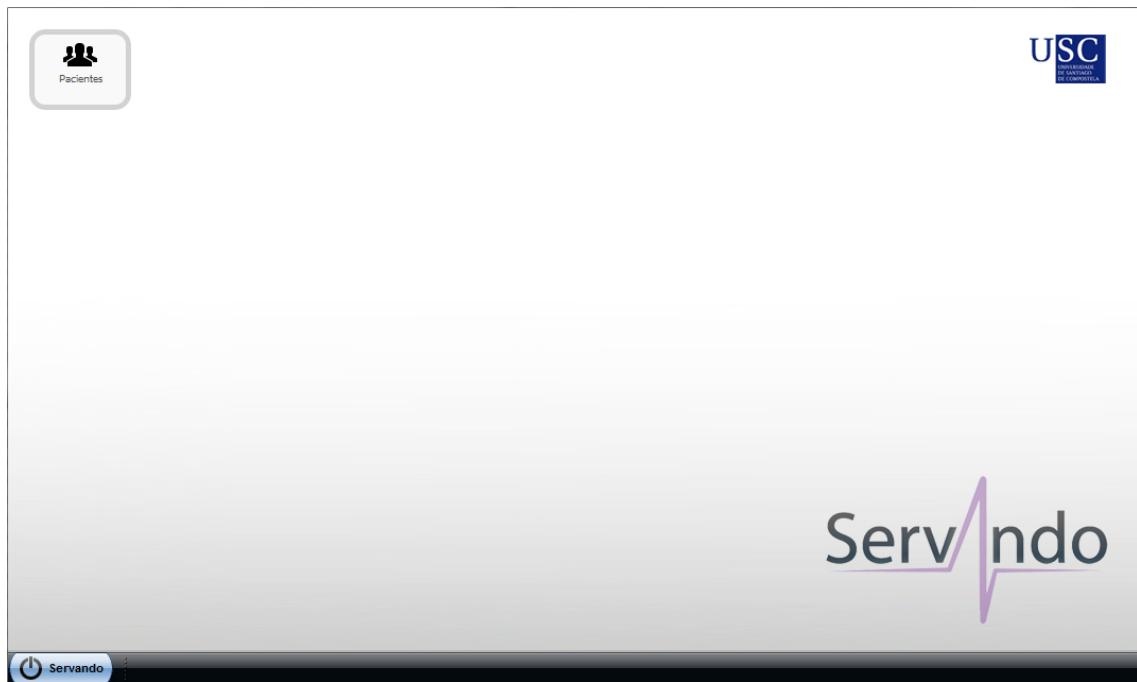


Figura 5.3: Interface da aplicación despois do inicio do sistema

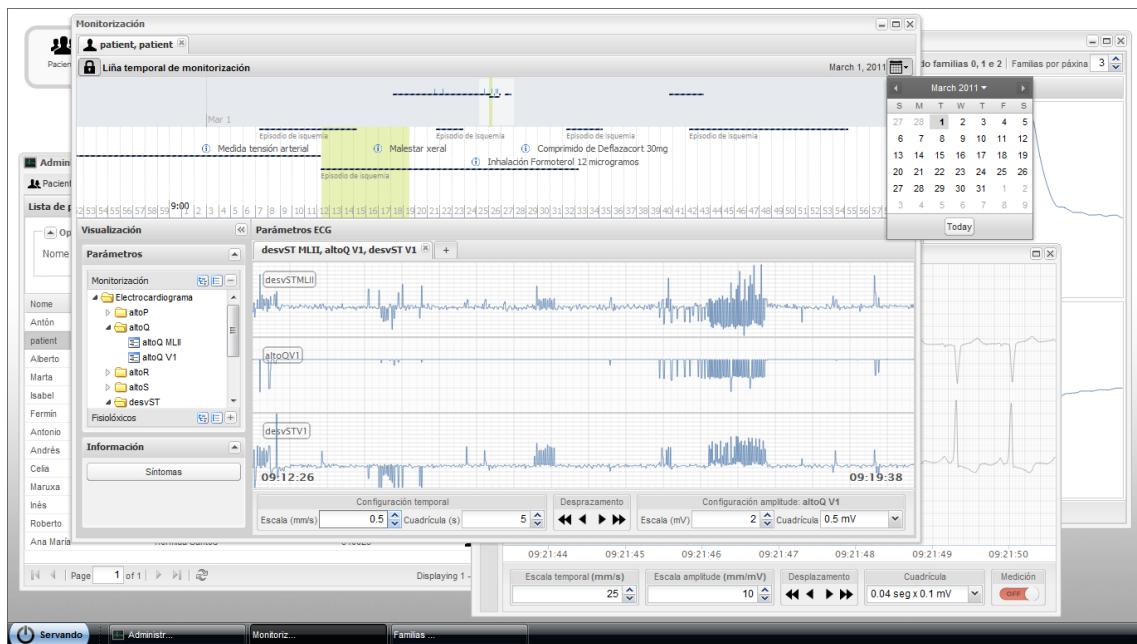


Figura 5.4: Interface da aplicación coas diferentes ventás dispoñibles

Ventá de seguimento

A ventá de seguimento engloba o conxunto de compoñentes que permiten a visualización do seguimento en distintos niveis de abstracción. A figura 5.5 mostra un prototipo coa súa estrutura. Na parte superior atópanse a liña temporal e o calendario, e abaixo a área de representación de parámetros. O resto de compoñentes que a conforman non están visibles directamente senón que se mostran cando é necesario para a representación de información de carácter máis específico. A continuación comentarase con máis unha delas.

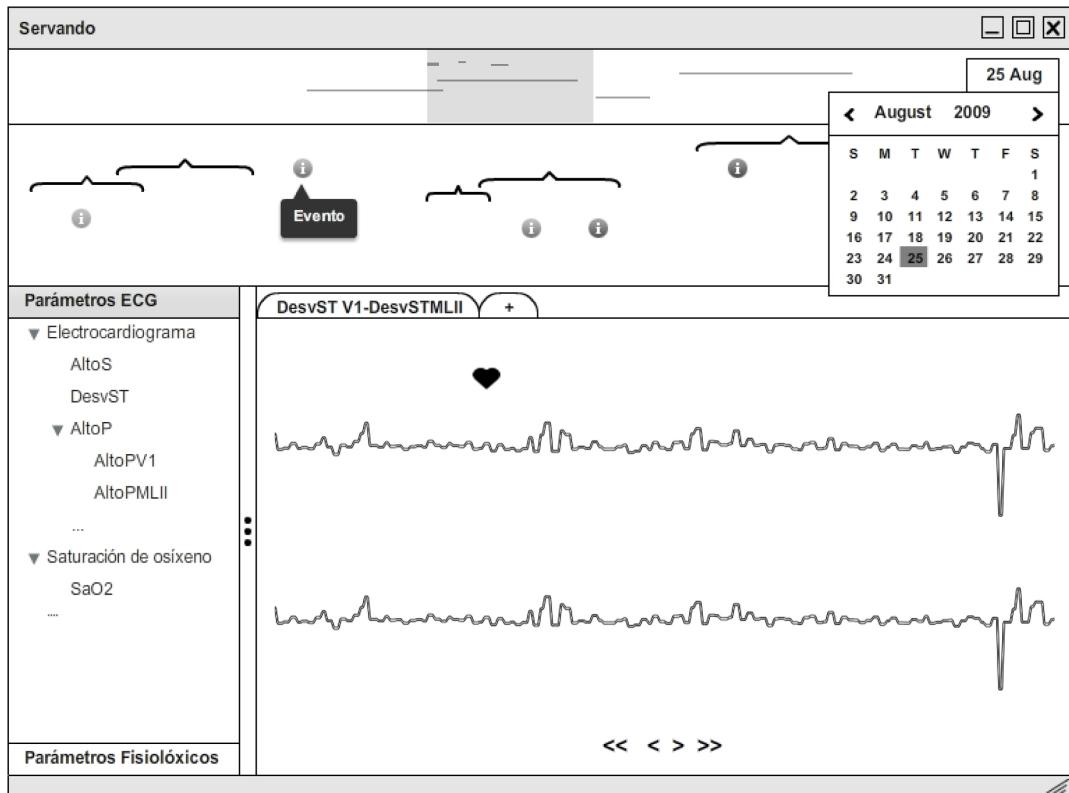


Figura 5.5: Etapas de seguimento dun paciente

Calendario

O calendario é a compoñente que permite que o usuario navegue a máis alto nivel polo seguimento dun paciente. A súa función principal é proporcionarlle información ó usuario sobre as datas nas que existen eventos, de xeito que este non se vexa obrigado a recorrer día a día a etapa de seguimento comprobándoo. Decidiuse que esta compoñente non estivera sempre visible, pois unha vez seleccionada unha data, o usuario céntrase na análise dos sucesos dese día ou do conxunto de

días que o rodean, perdendo o calendario a súa utilidade e ocupando un espacio considerable, polo que aparece integrado coa liña temporal de eventos que explicaremos a continuación.



Figura 5.6: Calendario

Liña temporal de eventos

A liña temporal de eventos é a compoñente da interface que proporciona unha maior abstracción, permitindo resumir unha gran cantidade de información, e mostrando únicamente aquela que é relevante. Nela represéntanse de xeito visual os diferentes eventos e episodios que teñen lugar durante o transcurso do seguimento, permitindo caracterizar de xeito rápido e sinxelo o ocorrido durnante un longo período de tempo. Como se ve na figura 5.7, está dividido en dous niveis:

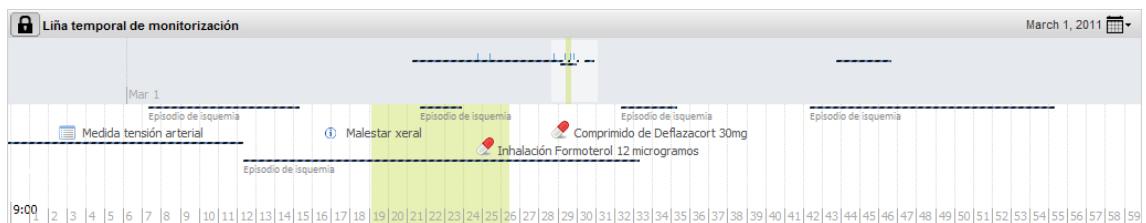


Figura 5.7: Línea temporal de eventos

- Un primeiro nivel no que se poden visualizar os eventos dunha franxa temporal de entre 1 e 15 días. Neste nivel os eventos móstranse como puntos e os episodios como liñas, permitindo que o usuario obteña un resumo dos sucesos detectados durante ese intervalo de tempo.
- Un segundo nivel, no que se permite ver ata un máximo de 24 horas. Neste nivel os eventos visualízanse como iconas acompañadas dun nome, variando a icona en función do tipo de evento. Os episodios represéntanse como franxas que marcan o intervalo de tempo no que se produciron. Esta banda móstrase únicamente o pasar o rato por enriba da outra, para non ocupar espacio, ánda que o usuario pode fixala, e facer que se manteña sempre visible. No parte central do nivel superior aparece resaltado nunha cor máis clara o intervalo de tempo que neste se visualiza, permitindo que o usuario relacione as dúas franxas.

Ó facer clic nun evento, amosaranse os detalles do mesmo nun cadro emerxente, mentres que ó facer clic nun episodio se mostrará na área de representación de parámetros o sinal asociado ó parámetro sobre o que o episodio foi detectado, permitindo obter de forma simple e rápida a súa información.

O usuario pódese desprazar temporalmente facendo clic e arrastrando a liña temporal, ou ben facendo dobre clic nun punto concreto da liña, co que a ventá temporal se centra no instante seleccionado. Isto é especialmente útil no primeiro nivel, pois ó visualizárense uns cantes días, resulta un xeito moi cómodo de desprazarse por eles. Tamén pode mover a liña temporal usando as frechas do teclado.

Para cambiar a escala temporal, o usuario debe facer *scroll* sobre a liña, en calquera dos dous niveis. Este xesto é usado na maioría de aplicacóns para efectuar cambios no nivel de detalle.

Cando se reciben novos eventos dende os dispositivos móbiles, estes visualízanse automaticamente na liña temporal. Decidiuse indicar este tipo de sucesos mediante notificacións como a que se ve na figura 5.8, no canto de desprazar a liña ata o momento do suceso, que podería resultar incómodo.

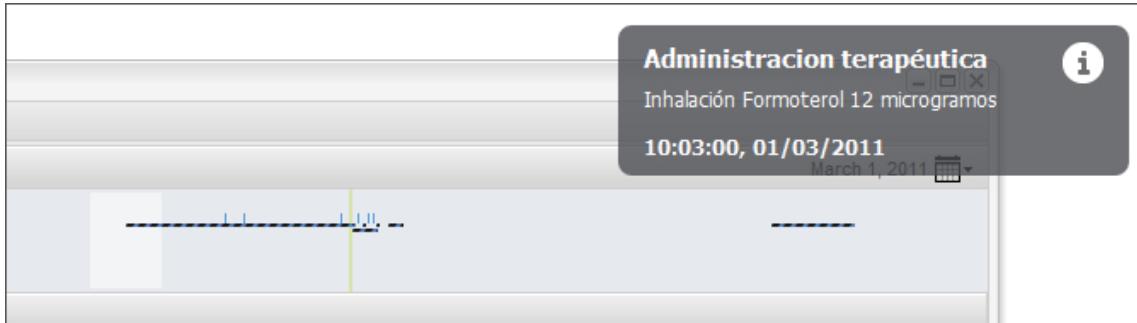


Figura 5.8: Notificación en tempo real dun evento

Área de representación de parámetros

Esta área da interface (figura 5.9) permite a visualización de parámetros fisiológicos e do diagrama de tendencias. Dado que os parámetros dun e doutro tipo se representan en escalas temporais moi diferentes decidiuse separar a súa visualización en dúas vistas que permitan obter a maior información posible de cada conxunto, utilizando para elo as escalas axeitadas en cada caso.

Como se pode ver na figura 5.9 está dividida en dúas zonas.

- Na zona esquerda móstrase un menú en forma de acordeón que permite acceder ás diferentes categorías de parámetros dispoñibles. En cada elemento do menú móstrase unha árbore cos diferentes parámetros dispoñibles organizados xerarquicamente en categorías. Este menú está deseñado de tal xeito que ó abrir unha das súas seccións a vista adáptase a natureza dos parámetros que nela se poden seleccionar, sen que o usuario teña que cambiar a vista manualmente mediante outro tipo de control. Esta zona está por defecto visible, aínda que

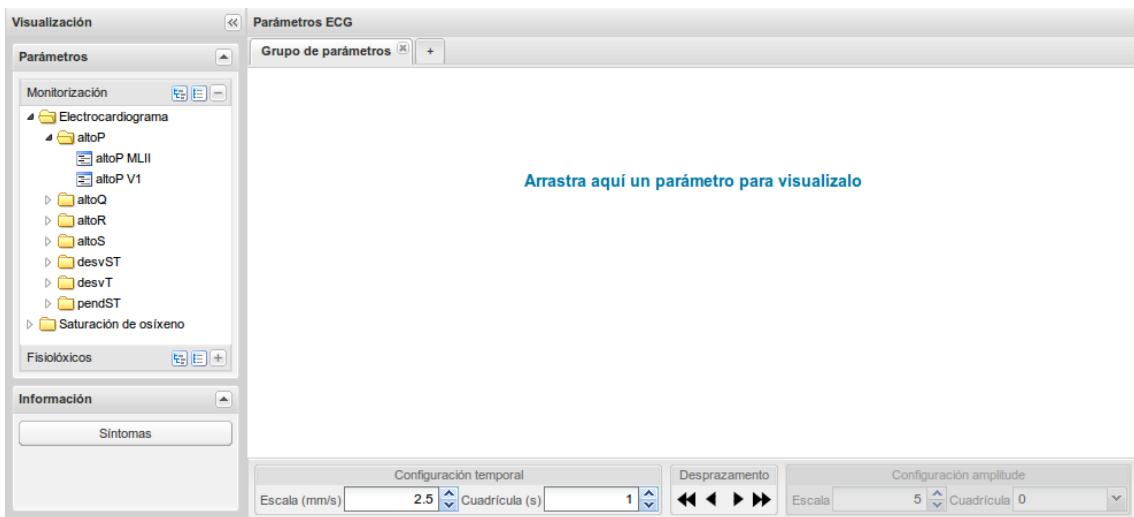


Figura 5.9: Área de representación de parámetros

o usuario pode repregala á marxe da pantalla para ter máis espacio. Neste estado, ó acercar o rato á marxe visualizase de novo a árbore de parámetros e en canto o usuario aparte o rato volverá agocharse.

- A zona da dereita é a área na que se representan os sinais dos diferentes parámetros. Está deseñada como un panel de pestanas, no que se poden ir creando grupos de parámetros que se visualizan conxuntamente.

Para seleccionar un parámetro e engadilo á visualización da pestana que está activa, non hai máis que facer dobre clic sobre el, ou arrastrar o nodo que representa ó parámetro na árbore de selección directamente á zona de visualización. Mientras se arrasta o parámetro, aparece xunto ó cursor unha icona que indica a posibilidade de soltalo nun lugar correcto, como se amosa na figura 5.10. Unha vez engadido axústanse as dimensións do resto de parámetros para repartir o espazo dispoñible. Para desprazarse temporalmente o usuario pode facelo mediante as frechas do teclado, ou usando os controis con forma de frecha da parte inferior.

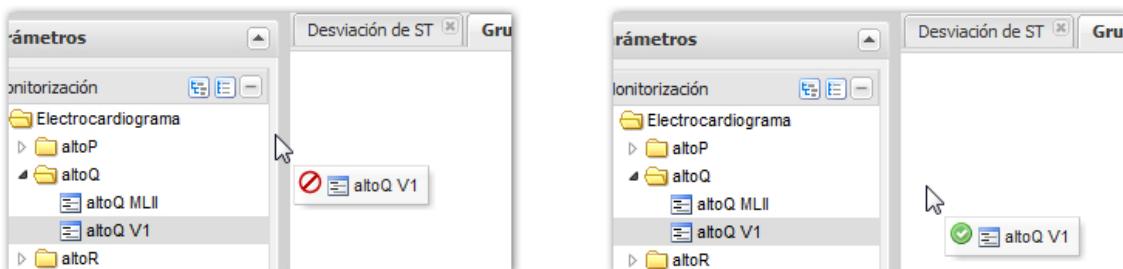


Figura 5.10: Estado do cursor durante a selección dun parámetro

Sobre cada un dos parámetros visualizados pódense realizar anotacións e medicións de xeito sinxelo. Pódese acceder ás correspondentes ferramentas mediante o menú de utilidades que se

mostra ó situar o rato enriba dun determinado parámetro (figura 5.12). Este mecanismo permite ter á man as ferramentas de edición sobre o sinal que se está analizando, sen que interfira en ningún momento coa visualización, e sen ter que acceder a terceiros menús situados normalmente a unha distancia considerable da zona na que se traballa que permitan facer uso delas.

- As anotacións permítenlle ao persoal clínico caracterizar con maior exactitude aqueles detalles que consideren preciso resaltar sobre o sinal dos diferentes parámetros. Para a realización de anotacións pódese facer clic nun punto, ou ben facer clic e arrastrar ata outro punto. Mediante a primeira opción realizaremos unha anotación dun punto concreto, e mediante a segunda anotaremos o intervalo temporal comprendido entre ámbolos dous puntos. Os dous tipos de anotación pódense ver na figura 5.11.

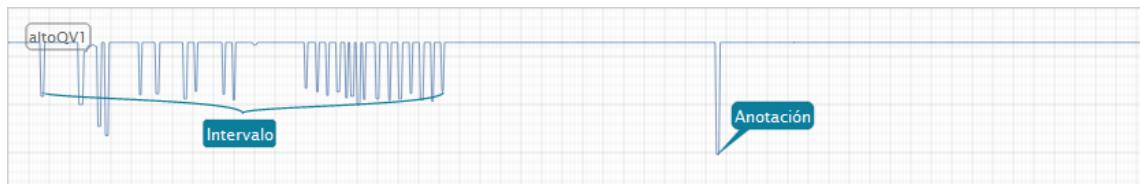


Figura 5.11: Anotacións

- A realización de medicións faise do mesmo xeito, seleccionando o punto de inicio e arrastrando ata o de fin. Por defecto, durante a medición, rastréxanse os puntos que máis preto están do cursor, aínda que esta opción se pode desactivar. Aínda que tamén é posible realizar medicións mediante o uso das escalas, esta funcionalidade permite facelo dun xeito moi simple e sen obrigar ó usuario a realizar interpretacións sobre a pantalla, que finalmente non reportan o mesmo nivel de exactitude. A figura 5.12 ilustra esta funcionalidade.



Figura 5.12: Realización dunha medición

A figura 5.13 amosa a vista de **diagramas de tendencias**, coa área de selección de parámetros repregada no marxe esquierdo da pantalla. Na parte inferior sitúanse os diferentes controis de escalas e cuadrículas, xunto cos botóns que permiten o desprazamento polo sinal. Os episodios detectados e os fragmentos de ECG obtidos durante a monitorización aparecen indicados sobre o sinal, os primeiros como unha zona remarcada e os segundos mediante unha icona.

A visualización de **parámetros fisiolóxicos** realiza en intervalos de ata 15 días, como se observa na figura 5.14. Para manter a consistencia entre os distintos niveis de información, e

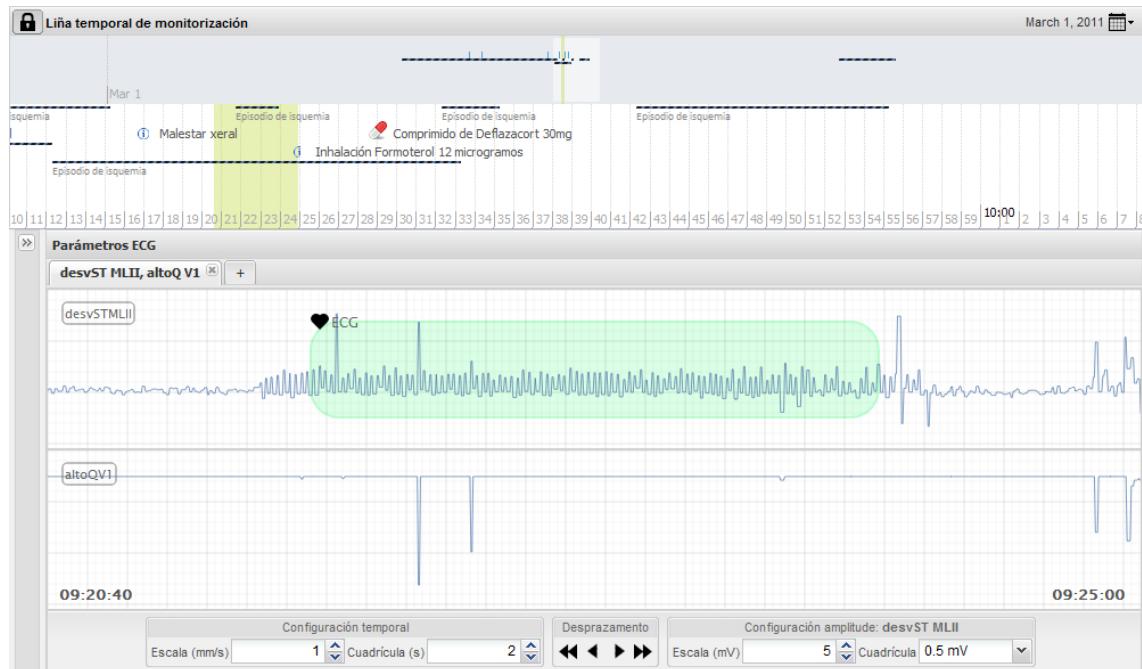


Figura 5.13: Vista de diagramas de tendencias

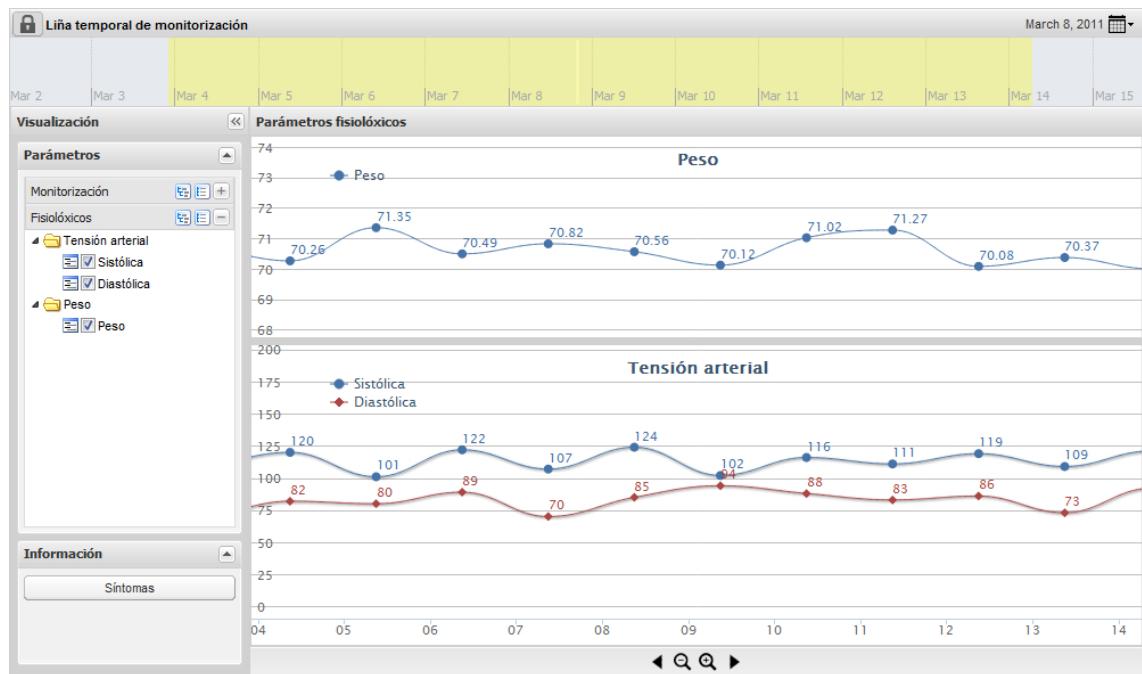


Figura 5.14: Vista de parámetros fisiológicos

manter a xerarquía proposta, a banda inferior da liña temporal ocúltase durante a visualización deste tipo de parámetros, pois mostraría un intervalo de tempo menor, que podería dar lugar a confusións, e rompería co modelo proposto.

Ventá de representación de ECG

A ventá de representación de ECG permite a visualización de sinais de electrocardiograma. Accédese directamente a ela seleccionando unha das marcas de ECG que se mostran na área de visualización de parámetros, coma a que aparece no primeiro parámetro da figura 5.13. Isto permite que o usuario acceda directamente ó fragmento do sinal sobre o que se identificou un suceso concreto ou algúm tipo de anomalía, sen este teña que indicar ese instante doutro xeito máis tarde.

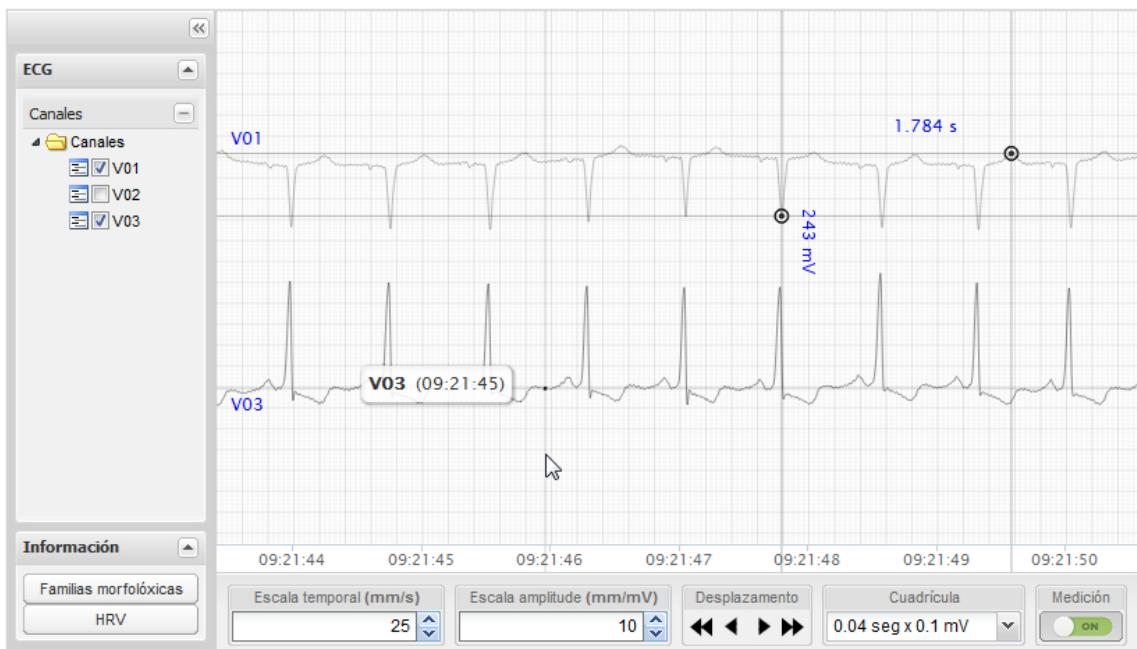


Figura 5.15: Ventá de representación de ECG

Na figura 5.15 pode verse o aspecto visual desta ventá. Á esquerda sitúase un menú no que se poden seleccionar as canles do electrocardiograma que se desexan visualizar. A área da dereita é usada para a representación do sinal das diferentes canles do ECG. Tanto o estilo dos menús como o dos diferentes controis é o mesmo que o utilizado no resto de compoñentes presentados.

Dende esta ventá pódese acceder á ventá de familias morfolóxicas que se mostra na figura 5.16 a través do botón situado na parte inferior esquerda. Dende esta ventá o usuario pode visualizar un resumo dos diferentes latidos dun paciente nun instante concreto de tempo, facilitando a identificación de sucesos anómalo. Por cuestiós de espacio dispoñible, pode resultar incómodo visualizar conxuntamente todas as familias dun determinado instante, por iso se decidiu organizalas en páxinas, polas que o usuario se pode desprazar mediante os controis da parte superior. En todo caso, é o usuario quen elixe o número de familias que quere ver en cada unha das páxinas, podendo

visualizalas todas xuntas se así o desexa.

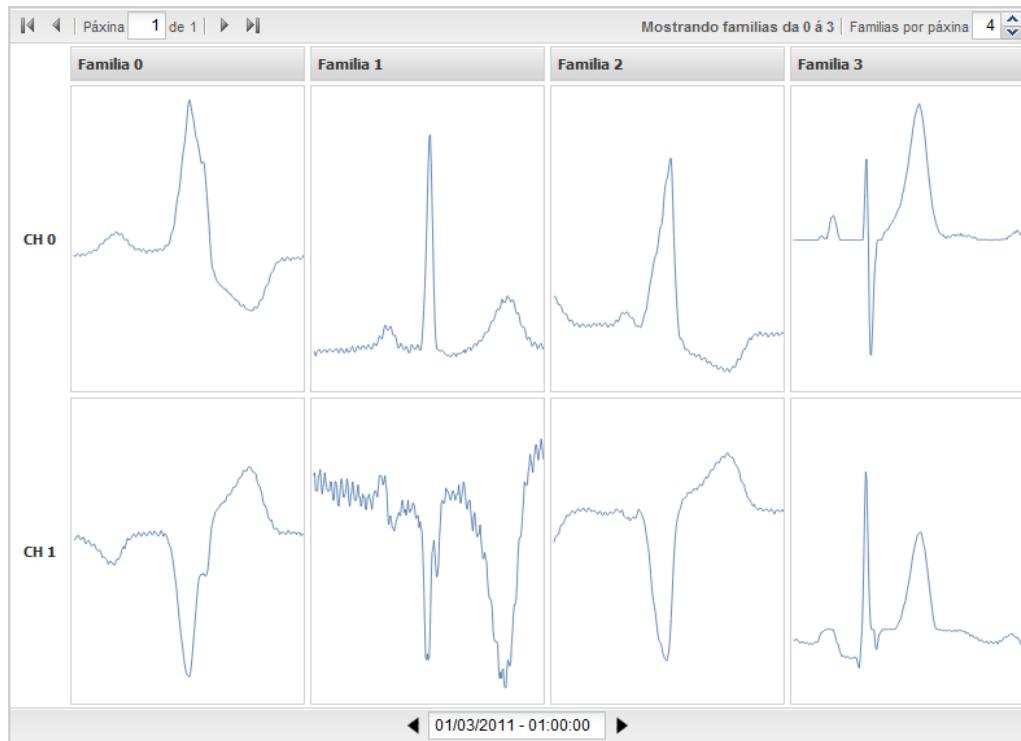


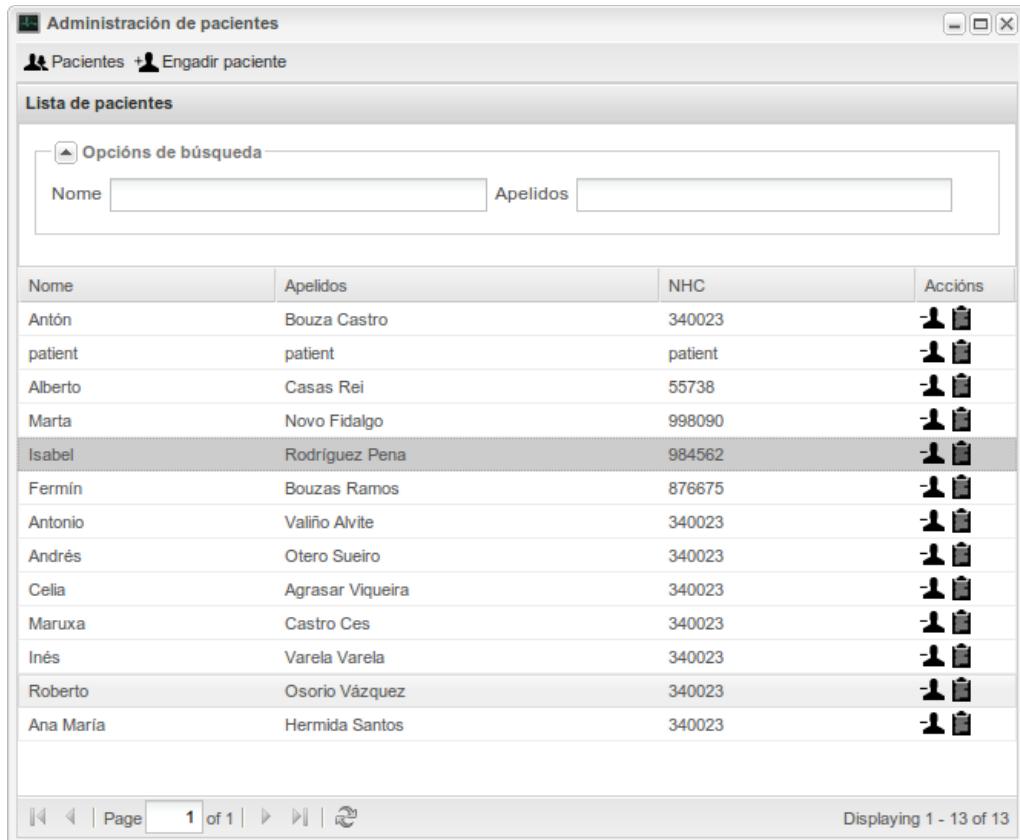
Figura 5.16: Ventá de representación de familias morfológicas

Na parte inferior móstrase o instante temporal das familias visualizadas e uns controis que permiten acceder ó seguinte ou anterior conxunto de familias respectivamente.

Ventá de administración de pacientes

A ventá de administración de pacientes permite o cumprimento de tó dolos requisitos relacionados coa xestión dos pacientes do sistema por parte do persoal clínico ou dun administrador do sistema. Accédese a ela a través do menú principal ou mediante o lanzador que aparece no escritorio. Como se pode ver na figura 5.17, unha vez abrimos esta ventá móstrase a lista de pacientes dados de alta no sistema, e dous campos de texto que nos permiten filtralos segundo os seus datos persoais. A busca realiza se a medida que o usuario escribe, e a lista vaise actualizando.

Ó seleccionar un paciente da lista accedemos ó seu perfil, e nel atopamos dúas pestanas. A primeira (figura 5.18) permitenos xestionar o conxunto de etapas de seguimento do paciente, e acceder á visualización do seu seguimento seleccionando o botón circular que se mostra despois da descripción de cada etapa. Na outra podemos editar os datos persoais do paciente.



Nome	Apelidos	NHC	Accións
Antón	Bouza Castro	340023	 
patient	patient	patient	 
Alberto	Casas Rei	55738	 
Marta	Novo Fidalgo	998090	 
Isabel	Rodríguez Pena	984562	 
Fermín	Bouzas Ramos	876675	 
Antonio	Valiño Alvite	340023	 
Andrés	Otero Sueiro	340023	 
Celia	Agrasar Viqueira	340023	 
Maruxa	Castro Ces	340023	 
Inés	Varela Varela	340023	 
Roberto	Osorio Vázquez	340023	 
Ana María	Hermida Santos	340023	 

|◀ |◀ | Page | 1 of 1 | ▶|▶| ⌂ | Displaying 1 - 13 of 13

Figura 5.17: Lista de pacientes

5.4. Deseño e implementación do servidor

No conxunto da aplicación, o servidor é responsable da persistencia da información e de ofrecer os servizos que permitan acceder a ela. Para isto, como vemos na figura 5.19, que se corresponde cunha parte da arquitectura do sistema, conta con tres compoñentes que implementan unha parte da funcionalidade requirida. Nesta sección comentaremos en primeiro lugar o deseño e implementación do **Motor de persistencia**, e a continuación o dos **controladores** e do **Adaptador JSON**. Os dous últimos explicaranse de xeito conxunto, dado que están estreitamente relacionados.

5.4.1. Motor de persistencia

Como se avanzou no capítulo 4, a persistencia do modelo presentado na sección 5.1 realiza mediante *Hibernate*. Isto require da definición dun esquema de tradución entre o modelo de datos co que traballamos, e o modelo relacional que utilizará a base de datos.

Para a definición dese esquema, utilízase a ferramenta *Hibernate Annotations*, que permite definir como se realiza a tradución dos diferentes atributos de cada clase a campos da base de datos, mediante a inserción das etiquetas apropiadas en cada un deles. As relacións entre as diferentes



Figura 5.18: Etapas de seguimento dun paciente

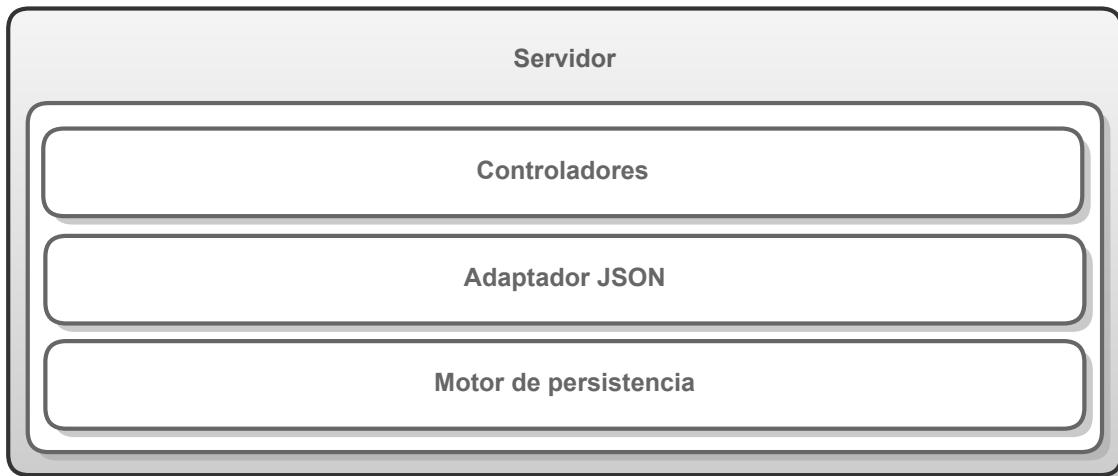


Figura 5.19: Arquitectura do servidor

clases anotadas (*entidades*) tamén se establece mediante anotacións.

As clases que representen entidades deben proporcionar nomes estándar *JavaBean* xunto con *setters* e *getters* para tódalas súas propiedades que vaian ser incluídas na base de datos, de xeito que *Hibernate* poida utilizarlos para construír os obxectos. Hai que subliñar que as propiedades simples (String, Integer, Float, etc.) non é obrigatorio anotállas, pois *Hibernate* fai o mapeo directamente ós correspondentes tipos de datos SQL, aínda que se quere especificar un xeito diferente de facelo está permitido. Tamén deberemos proporcionar un ficheiro de configuración *persistence.xml* no que

se establecen os datos de acceso á base de datos e a lista de clases que se xestionarán. Neste ficheiro podemos indicarlle a *Hibernate*, entre outras moitas opcións, que en caso de non existiren táboas na base de datos, as cree automaticamente, ou engada o aquelas necesarias.

Unha vez temos as clases anotadas, podemos utilizar as clases *EntityManagerFactory* e *EntityManager* que proporciona a API de *Hibernate* para acceder á información da base de datos, engadila e modificala.

Neste caso, fíxose uso delas para construír un conxunto de DAOs [15] que permiten unha xestión transparente do acceso á base de datos. Estas clases, mostradas na figura 5.20, apóianse na funcionalidade ofrecida por *Hibernate* para realizar consultas e ofrecer métodos concretos de busca, gardado e borrado para cada tipo de dato do modelo. O acceso elas unificouse na clase **DAOFacade** utilizando o patrón de deseño *facade* proporcionando así un punto único de entrada ós mecanismos de almacenamento e xestión de datos.

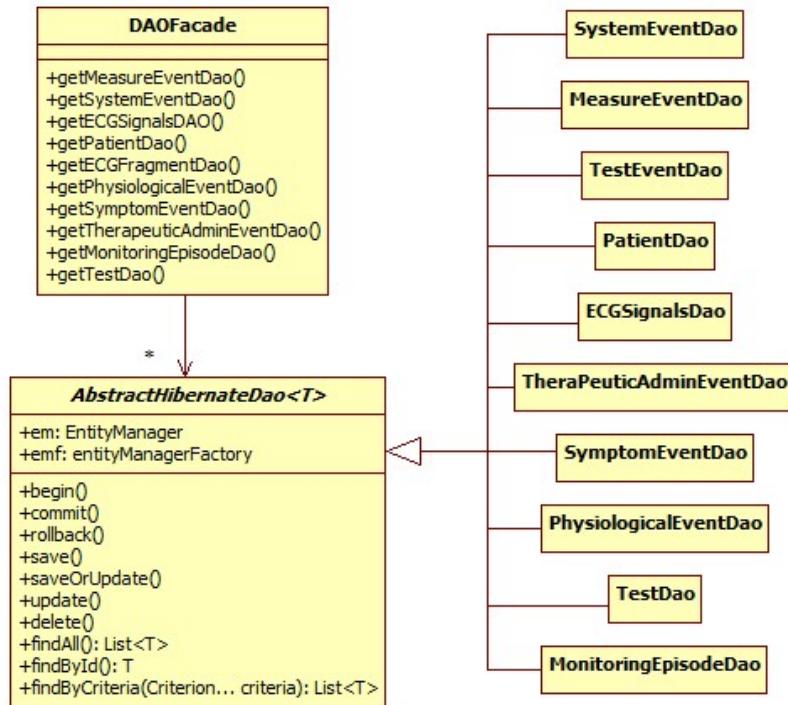


Figura 5.20: Clases de acceso datos

Por outra banda, para almacenar de xeito eficiente o conxunto de valores puntuais das distintas sinais recollidas, faise uso de cambios binarios (*blobs*). A medida que se van recibindo valores dun determinado sinal, estes tradúscense a *arrays of bytes* e engádense ós recibidos con anterioridade no correspondente *blob* na base de datos. A xestión deste tipo de datos non pudo automatizarse directamente mediante *Hibernate*, pois o acceso a un fragmento concreto dun sinal require a realización de tarefas complexas como o cálculo de desprazamentos a partir da súa frecuencia. Polo tanto, creouse a clase *BlobReference*, que permite xestionar este tipo de datos de forma totalmente

transparente mediante os métodos `getBytes()` e `setBytes()`, que permiten obter fragmentos do sinal sen cargalo totalmente en memoria. Esta estratexia está totalmente integrada nos diferentes DAOs, e o manexo de clases que teñan atributos deste tipo é idéntico ó do resto de clases que xestiona *Hibernate*. completo.

5.4.2. Controladores e Adaptador JSON

Como xa se comentou durante a descripción da arquitectura do sistema, a transferencia de información entre o servidor e o cliente realizarase utilizando o formato de intercambio de datos *JSON*. Para levar a cabo esta tarefa deseñouse o conxunto de clases que se mostran na figura 5.21.

Por unha banda creáronse dúas clases que actúan como controladores, *PatientController* e *MonitoringController*, cada un dos cales é responsable de xestionar un subconxunto das diferentes peticionés recibidas dende o cliente. O primeiro deles atende tódalas peticionés relacionadas coa xestión de pacientes, mentres que o segundo atende tódalas peticionés relacionadas coa solicitude de información de seguimento. As dúas herdan de *AbstractController*, que prové métodos de xestión da sesión, validación de parámetros, autenticación da solicitude, e envío de respuestas adecuadas a partires de obxectos *JSONObject*.

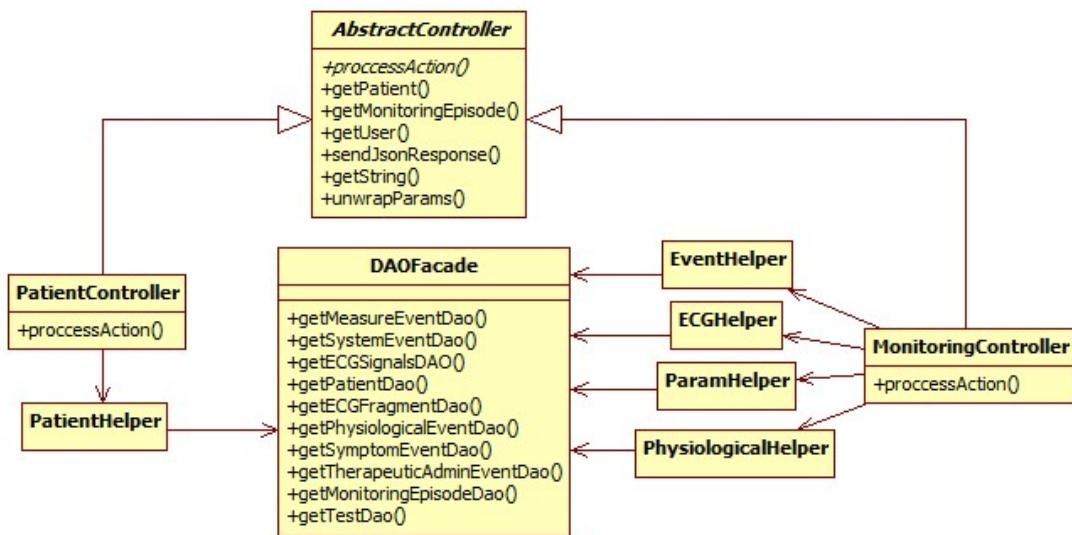


Figura 5.21: Controladores e adaptador *JSON*

Ámbolos dous se apoian nun conxunto de *helpers*, encargados da obtención da información da base de datos utilizando a fachada de acceso a datos comentada no apartado 5.4.1 e da súa tradución ó formato *JSON*. Esta tradución produce como resultado un obxecto *JSONObject*, co que o controlador responde á petición mediante o método `sendJsonResponse()` que proporciona a clase *AbstractController*, enviándolle ó cliente un fragmento de texto *JSON* cos datos solicitados.

A continuación móstrase un exemplo de tradución dun obxecto da clase *EpisodeEvent* que

referencia un fragmento de ECG ó formato *JSON* para o seu posterior envío ó cliente.

```
{
  id: 9986,
  tag: 'isquemia',
  param: 'desvST',
  lead: 'MLII',
  timestamp: '20110301204356',
  endTimestamp: '20110301204524',
  description: 'Episodio de isquemia',
  ecgFragment:{  
    rate: 4,  
    leads:[{  
      name: 'V1',  
      values: [987,986,987,985,945, ...]  
    },{  
      name: 'V2',  
      values: [987,986,987,985,945, ...]  
    }]  
  }
}
```

5.5. Deseño e implementación do cliente

Nesta sección partiremos da arquitectura proposta no capítulo 4 para comentar o deseño e implementación dos diferentes componentes do cliente, que se mostran na figura 5.22.

5.5.1. Núcleo do cliente

Dentro da capa cliente, a parte que ten unha maior relevancia é o **núcleo**. Na figura 5.23 móstrase o conxunto de clases deseñado para dar soporte o resto de módulos e realizar as función principais desta parte da arquitectura. A continuación describense as más importantes:

- A clase *Application* é a clase máis importante do cliente. Nela recae a responsabilidade de cargar os diferentes módulos do sistema, e serve de nexo entre eles e o resto de componentes como o xestor de notificacións ou o xestor do escritorio. Durante o inicio da aplicación esta clase é a encargada de solicitar os datos do usuario, validalos, e posteriormente iniciar ou rexistrar os módulos do sistema. Hai que facer énfase no de rexistar, pois realmente o único que se fai é engadir os correspondentes lanzadores ó escritorio e ó menú principal, e non se cargan en memoria ata que o usuario necesita fazer uso deles.
- A implementación dos diferentes módulos realizaase estendendo a clase *Module*. Esta clase proporciona os métodos necesarios para permitir que a clase *Application* inicie os diferentes

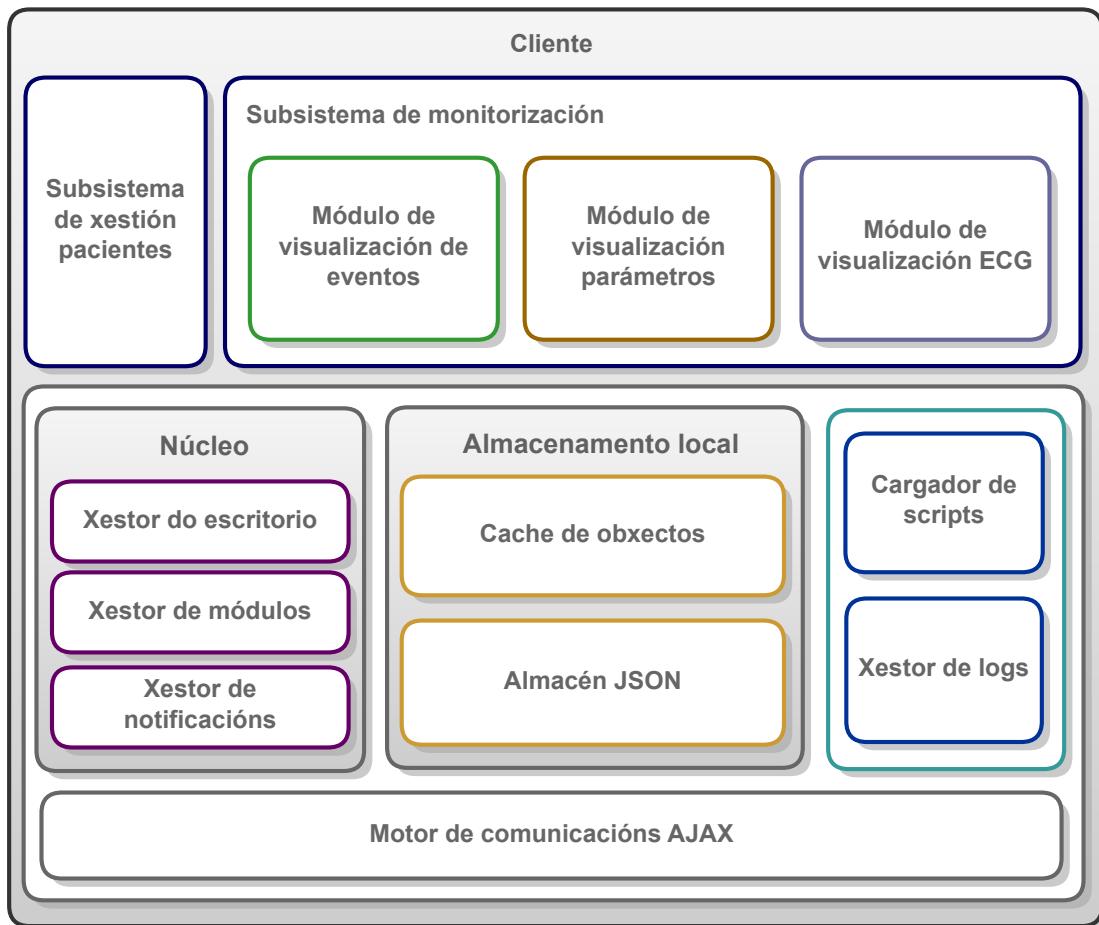
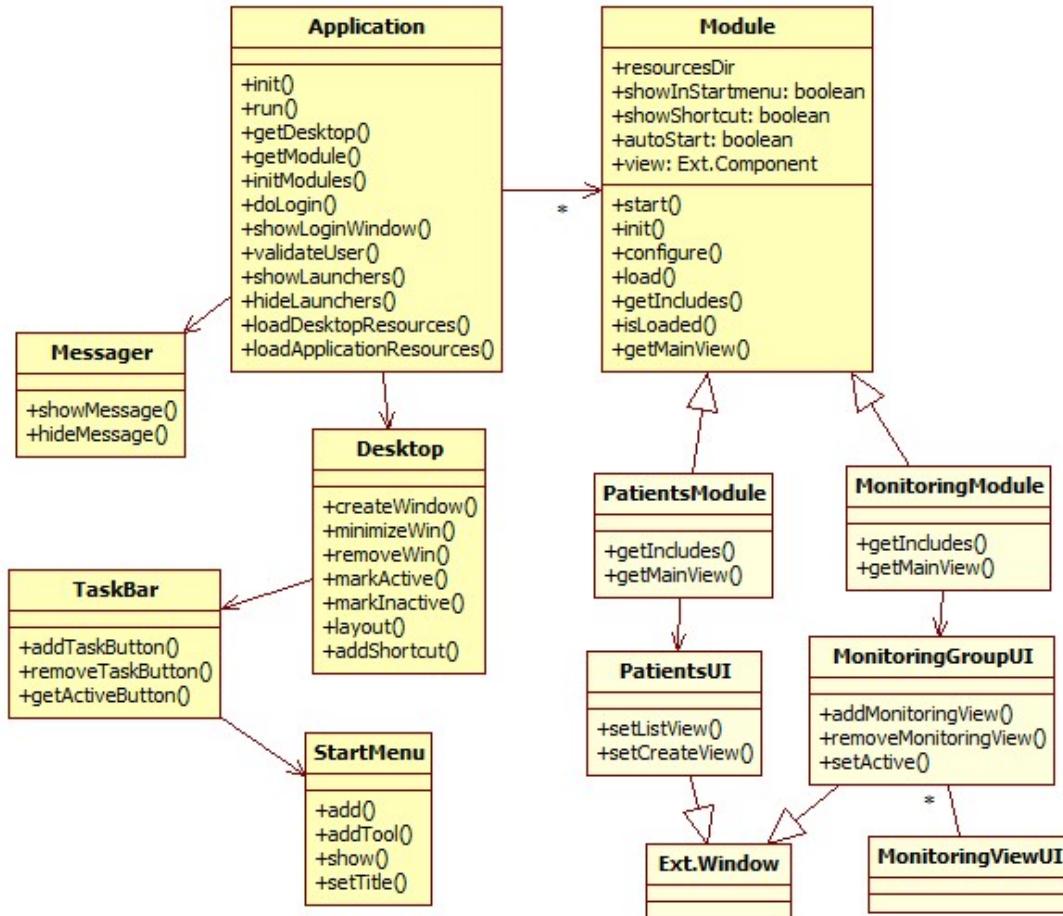


Figura 5.22: Arquitectura do cliente

módulos correctamente, cargue os recursos necesarios para o seu funcionamento, e configure o módulo segundo as súas preferencias. Pola súa parte, as clases que a estenden deben implementar os métodos `getIncludes()` e `getMainView()` que devolven a lista de `scripts` e o compoñente gráfico que implementa a vista principal do módulo respectivamente. A propiedade `autoStart`, que por defecto ten valor `false`, indica se o módulo debe ser cargado durante o arranque da aplicación, e é útil para realizar a precarga daqueles módulos más frecuentemente utilizados ou necesarios para o funcionamento do sistema.

Unha vez que se iniciou a aplicación e se rexistraron os diferentes módulos, a secuencia de carga dun módulo é a seguinte:

1. Cando se solicita o acceso a un módulo concreto a aplicación chama ó seu método `start()`.
2. O módulo comproba se os seus recursos están dispoñibles chamando ó método `isLoaded()`. Se o o están, realiza unha chamada ó seu método `init()` que inicia o proceso de arranque do módulo. Se non o está, realiza unha chamada ó método `load()`, que se encarga de cargar os recursos necesarios para o seu funcionamento. Esta carga realiza-

Figura 5.23: Diagrama de clases do **núcleo** do cliente

en segundo plano e a función *configure()* execútase como *callback* cando se completa a operación.

3. Unha vez cargados os recursos e configurado o módulo, o propio módulo fai unha chamada ó método *init()* para iniciarse.

Esta secuencia de operacións, que se mostra en forma de diagrama de secuencia na figura 5.24, permite retardar a carga dos diferentes elementos necesario para o funcionamiento dun módulo ata o momento do seu uso, reducindo a carga de memoria que se produce no navegador do usuario, optimizando o uso de recursos, e permitindo un inicio máis rápido da aplicación ó cargar durante o arranque unicamente aqueles módulos que son necesarios. Esta tarefa realiza de xeito totalmente transparente, pois a clase *Application* unicamente realiza unha chamada ó método *start()* e é o propio módulo o que se encarga de xestionar o resto do proceso.

- A clase *Desktop* permite a xestión das diferentes ventás que poidan estar presentes no es-

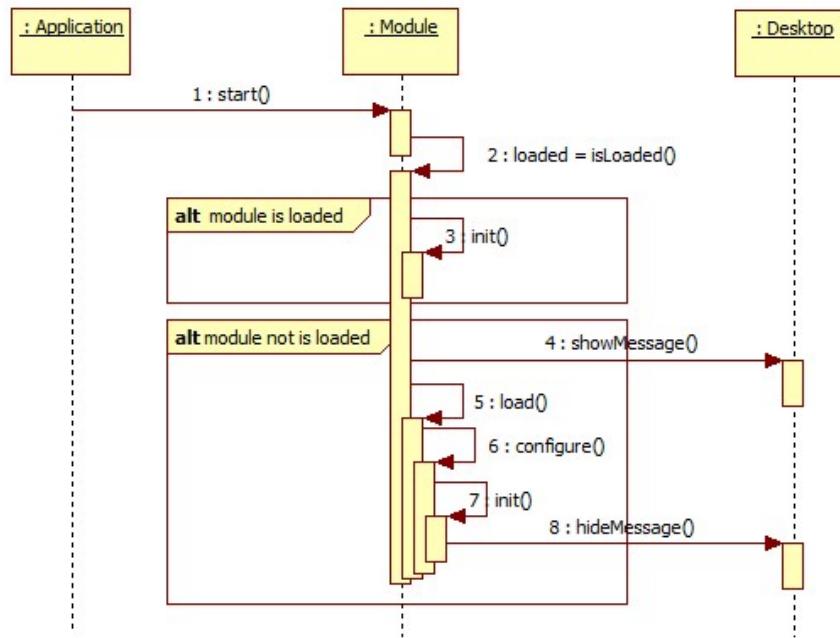


Figura 5.24: Diagrama de secuencia da carga dun módulo

citorio. Os diferentes módulos poden crear as súas propias ventás e xestionalas de xeito autónomo, ou ben poden delegar esta xestión en dita clase. Para iso, cada un deles deberá utilizar o método *createWindow()* que ofrece a clase *Desktop* para a creación das ventás, que as dota de funcionalidade extra como a capacidade de mostrarse na barra de tarefas, maximizarse, minimizarse, etc. Tamén ofrece o método *addShortcut()* que permite engadir lanzadores ó escritorio da aplicación.

Esta clase actúa en conxunto coas clases *Taskbar* e *StartMenu* que xestionan o menú principal e a barra de tarefas respectivamente.

- Por último, a clase *Messager* ofrece os métodos *showMessage()* e *hideMessage()* que permite mostrar notificacións no escritorio. Esta clase é usada, por exemplo, polos diferentes módulos para mostrar unha mensaxe de espera mentres se realiza a carga dos seus recursos.

5.5.2. Subsistema de monitorización

No subsistema de monitorización impleméntase a funcionalidade que permite a representación de toda a información obtida durante o seguimento dos pacientes. Como se pode ver na figura 5.25, divídese loxicamente en tres módulos.

Como se ve na imaxe, están relacionados entre si mediante a clase *MonitoringViewUI*, que representa a vista do seguimento dun usuario concreto dentro da aplicación.

Como se comentou no apartado 4.3, para a construción dos diferentes componentes fíxose unha

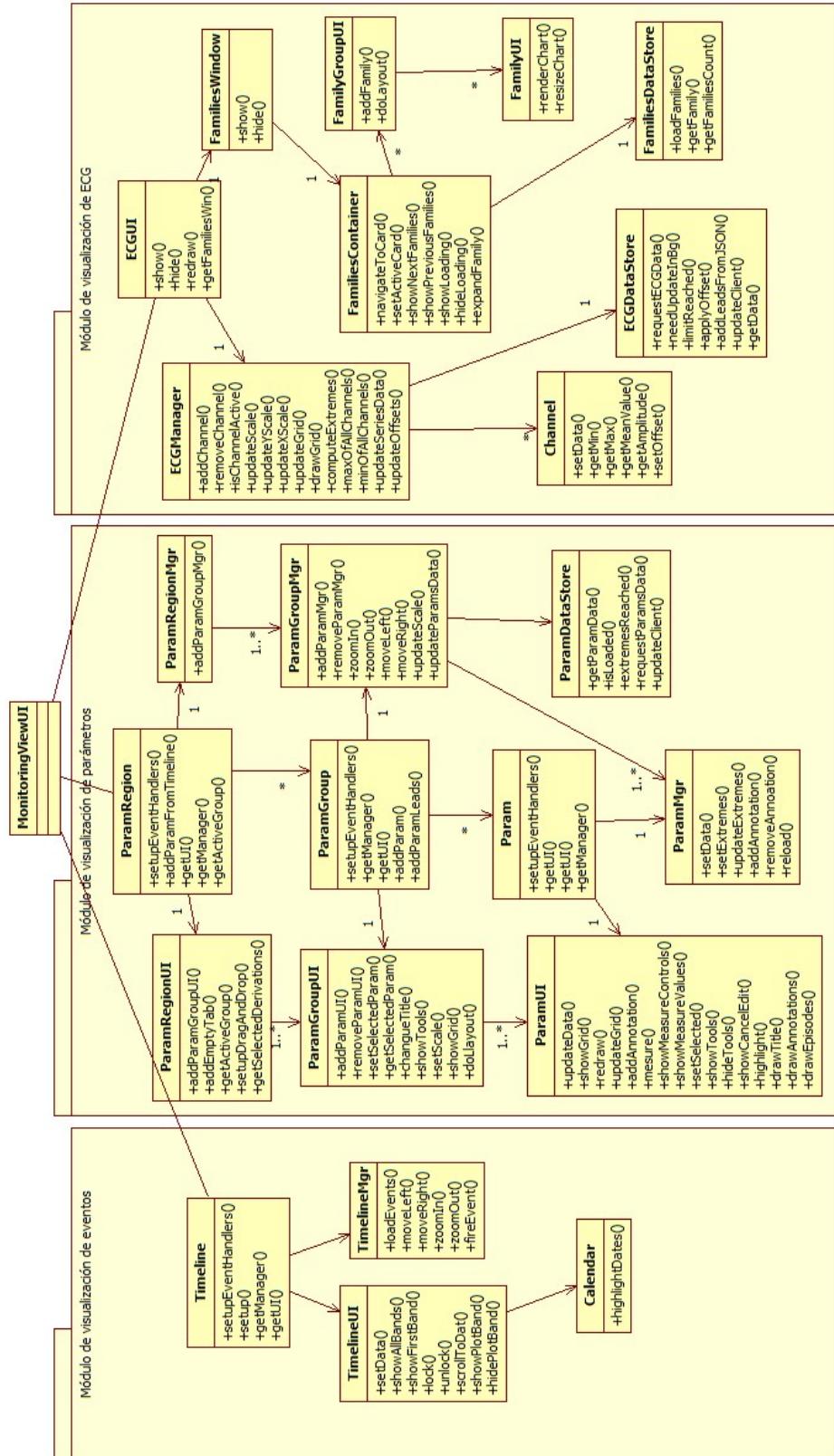


Figura 5.25: Diagrama de clases do subsistema de monitorización

división entre presentación e control utilizando os patróns *Observer* e *MVC*, creando xeralmente tres clases para cada compoñente individual, que representan a vista, o modelo e o control, agás naqueles casos que a lóxica necesaria é tan reducida que se decide incluíla na vista. A continuación explicaremos este proceso en detalle para o caso do módulo de representación de eventos e posteriormente comentaremos a función do resto de clases do diagrama da figura 5.25.

- **O módulo de representación de eventos** implementa a funcionalidade relativa á visualización de eventos mediante a liña temporal. As clases que permiten o seu funcionamento son as seguintes:
 - A clase *TimelineUI* representa o compoñente gráfico no que se visualizan os eventos en forma de liña temporal, e ofrece métodos para actualizar o conxunto de datos que se mostran, desprazarse temporalmente, e outras opción de modificación da vista. Esta clase apóiaise na clase *Calendar* para mostrar un calendario no que é posible resaltar un conxunto de datas concretas.
 - A clase *TimelineMgr* é responsable da lóxica de representación, é dicir, encárgase de procesar os diferentes eventos da interface, tratar correspondentemente os datos, e actualizar a vista.
 - A clase *Timeline* ten unha dobre funcionalidade. A primeira é a de separar a vista da lóxica, sendo a encargada de configurar a os manexadores dos diferentes eventos que se lanzan tanto dende a vista coma dende o control. Deste xeito a vista e o control limitánse a lanzar eventos que son atendidos polos seus correspondentes manexadores, pero entre eles un non sabe da existencia do outro. A segunda funcionalidade da clase *Timeline* é a de instanciar as dúas clases antes mencionadas no momento en que é necesario o seu uso, permitindo atrasar esta tarefa e aforrando os recursos necesarios para facelo.
- **O módulo de representación de parámetros** permite a visualización de parámetros fisiológicos e do diagrama de tendencias, e está formado polo conxunto de clases que se comentan a continuación:
 - As clases *ParamRegion*, *ParamRegionUI* e *ParamRegionMgr* encárganse da xestión da área de representación de parámetros. A súa labor e permitir a selección de parámetros e a creación de novos grupos deles, así como proporcionar métodos que permitan que o resto de compoñentes externos interactúen con esta parte da interface.
 - As clases *ParamGroup*, *ParamGroupUI* e *ParamGroupMgr* encárganse da visualización e xestión dun grupo de parámetros concreto. Permiten o manexo conxunto das escalas e cuadrículas dos diferentes parámetros visualizados, e proporcionan métodos para engadir e eliminar parámetros do grupo.
 - As clases *Param*, *ParamUI* e *ParamMgr* son responsables da visualización dun parámetro concreto. Permiten o manexo de escalas e cuadrículas de xeito individual, a realización de medidas e a xestión de anotacións.
 - A información relativa ós diferentes parámetros almacénase nunha cache local implementada na clase *ParamDataStore*. O funcionamento desta cache é o seguinte:

- Cando se solicita un fragmento de sinal dun parámetro concreto compróbase se este se atopa no almacenamento local.
- En caso de que non estea, solicítase ó servidor ese intervalo máis un determinado marxe por cada extremo que garanta a posibilidade de desprazarse polo sinal.
- A medida que o usuario se despraza polo sinal e se acerca a un dos extremos solicítanse novos datos en segundo plano, para evitar que este teña que esperar pola recarga no momento en que chegue ó límite.
- Outra función desta clase é a de realizar a mostraxe do sinal que se visualiza, de xeito que non se representen máis puntos dos que é posible mostrar, atendendo á resolución da pantalla.

Na figura 5.26 móstrase un diagrama de secuencia que representa a a selección dun parámetro para ser visualizado. Neste caso a acción e iniciada ó seleccionar un episodio na liña temporal.

■ **O módulo de representación de ECG**, que ten como obxectivo a visualización do sinal de electrocardiograma e das familias morfolóxicas, está constituído polas seguintes clases:

- A clase *ECGGUI* xunto coa clase *ECGManager* fan posible a representación do sinal de electrocardiograma nunha ventá, permitindo a selección das canles a visualizar, a xestión de escalas e cuadrículas. A clase *ECGGUI* permite ademais ter acceso á ventá de representación de familias morfolóxicas.
- As clases *FamiliesWindow*, *FamiliesContainer*, *FamilyGroupUI* e *FamilyUI* permiten a representación das familias morfolóxicas. A clase *FamilyGroupUI* representa a vista dun grupo de familias que se visualizan conxuntamente, e a clase *FamilyUI* representa a vista dunha única familia. Dado que a lóxica que se precisa para o manexo desta información é moi pouca, non se fixo necesario a creación de clases que realicen esa función, e a vista encárgase do control.
- A clases *ECGDataStore* e *FamiliesDataStore* representan os almacéns locais nos que se garda a información de electrocardiograma e familias morfolóxicas respectivamente. No caso do almacenamento da información de electrocardiograma, a clase *ECGDataStore* implementa un mecanismo de cacheado idéntico ó explicado para o caso do almacenamento da información de parámetros, que permite que o usuario se despraze polo sinal sen ter que esperar a que se cargue.

5.5.3. Subsistema de xestión de pacientes

O subsistema de xestión de pacientes é o encargado de implementar a funcionalidade que permite a administración dos pacientes dados de alta no sistema. As clases que o conforman, representadas na figura 5.27, son as seguintes:

- A clase *PatientsUI* representa a ventá de xestión de pacientes, e ofrece métodos que permiten cambiar a vista que se mostra, facendo posible cambiar entre a lista de pacientes ou a vista de creación de pacientes.

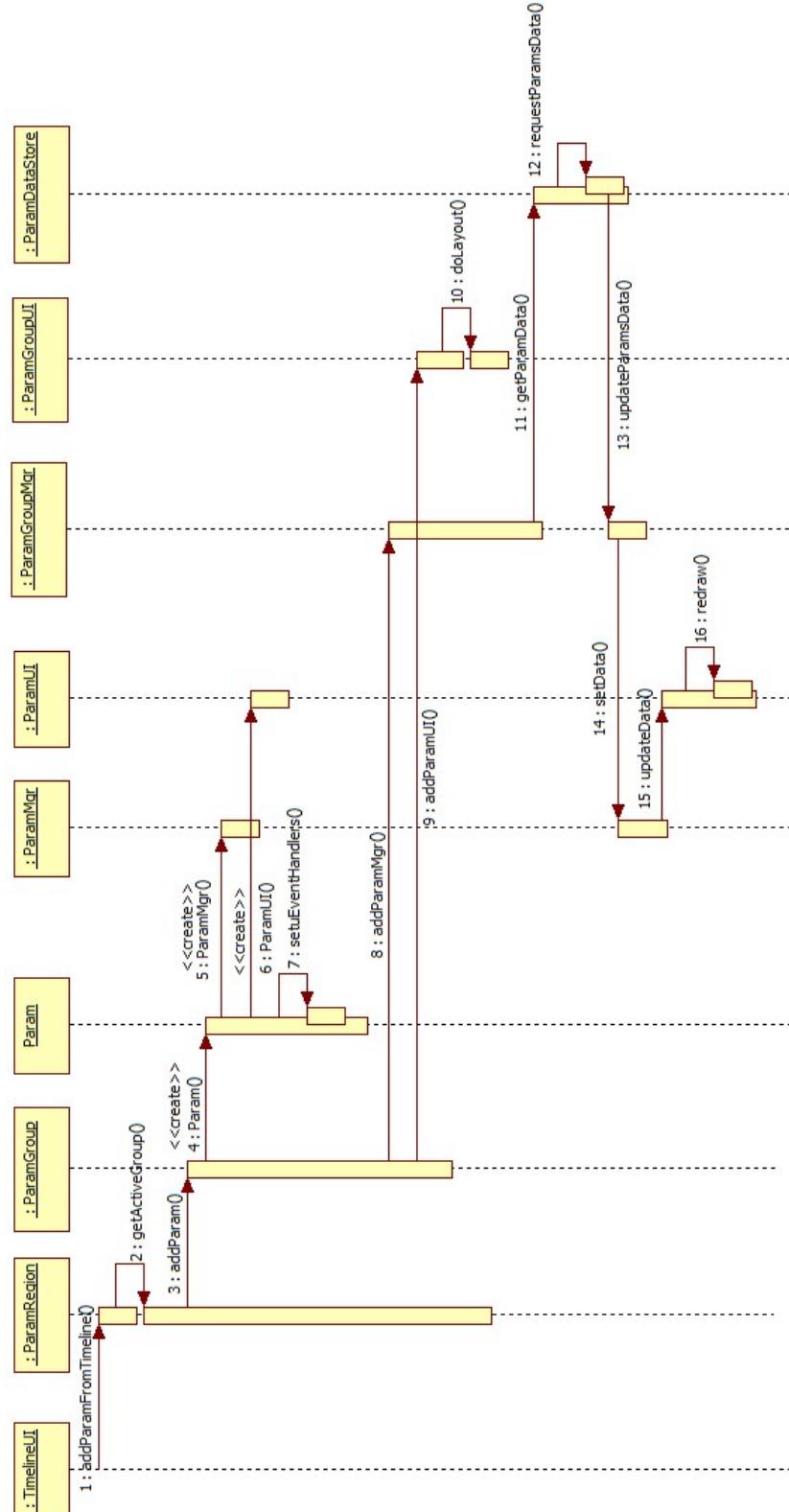


Figura 5.26: Diagrama de secuencia da selección dun parámetro

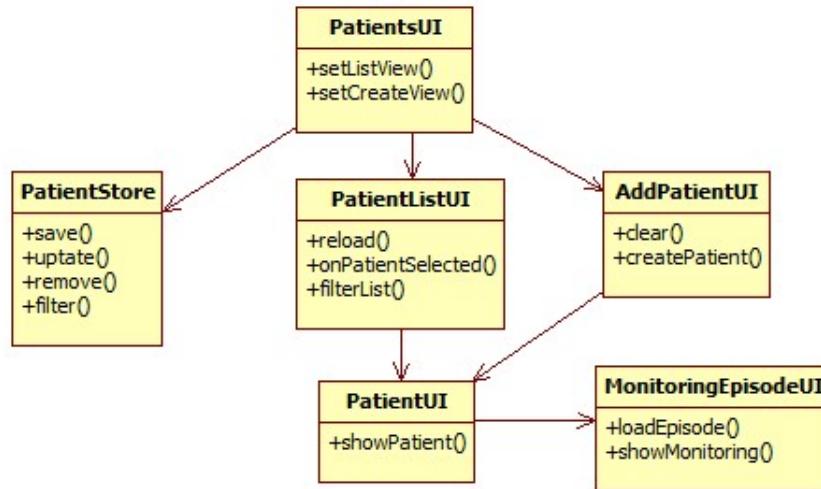


Figura 5.27: Diagrama de clases do subsistema de xestión de pacientes

- A vista da lista de pacientes impleméntase na clase *PatientListUI*. Esta permite cambiar á vista dun determinado paciente cando se selecciona, e proporciona o método *filter()* que permite filtrar os pacientes da lista.
- As clases *PatientUI* e *MonitoringEpisodeUI* representan a vista dun paciente e dunha etapa de seguimento concreto.
- O almacenamento local dos pacientes do sistema realiza na clase *PatientStore*. Neste caso, dado que a lóxica necesaria para o funcionamiento desta parte do sistema se limita á comunicación de operacións de persistencia ó servidor, esta clase é a encargada de realizaras, comunicándose a vista directamente con ela.

5.5.4. Módulos auxiliares

Os outros dous módulos presentados na arquitectura da figura 5.22 foron o **xestor de logs** e o **cargador de scripts**. Para a implementación do cargador de *scripts*, creouse a clase *ScriptLoader*, que ofrece o método *loadFiles()*, que permite cargar un *script* ou un conxunto deles en tempo de execución. Un dos parámetros da función é outra función que será executada como *callback* unha vez cargado o conxunto.

En canto ó **Xestor de logs** decidiuse utilizar a ferramenta *Log4Javascript* [3], deseñada específicamente para esta tarefa.

Capítulo 6

Probas e validación

Neste capítulo procederase á validación do sistema, realizando unha serie de probas que permitan demostrar a validez dos requisitos e restricións presentados no capítulo 3. Comezaremos describindo o conxunto de datos co que traballaremos e o entorno no que levaremos a cabo as probas, para a continuación expoñer o conxunto de tests que se deberán levar a cabo para validar os diferentes requisitos. A continuación mostraremos a avaliación heurística realizada para garantir a usabilidade da interface gráfica do sistema e finalmente realizaremos a validación dos diferentes requisitos.

6.1. Preparación das probas

Por cuestións operativas non foi posible ata o momento de redacción deste documento realizar a instalación da ferramenta no *Complexo Hospitalario Universitario de Santiago de Compostela*. Para a realización das probas, simulouse o entorno real instalando todo o software necesario para o funcionamiento do sistema nun ordenador persoal, isto é, un servidor de aplicacións (neste caso usouse un servidor *Tomcat*) e un servidor de bases de datos, que como xa se comentou no capítulo 4 será PostgreSQL. Para comprobar o funcionamento do sistema con eventos en tempo real, implementouse un *servlet* mediante o que se simulou a recepción de eventos remotos realizando múltiples chamadas dende un navegador web.

Os datos para a realización das probas foron subministrados polo cliente, e son datos de seguimento de pacientes reais monitorizados con Servando. Deste xeito comprobaremos tamén a compatibilidade coa plataforma sen necesidade de facer probas nun entorno de producción. O datos proporcionados polo cliente foron:

- Un conxunto de eventos de síntomas, administracións terapéuticas, medicións puntuais, etc. de varios días duración.
- 10 horas de sinal de 14 parámetros do diagrama de tendencias diferentes e de saturación de oxíxeno, que dada a ventá temporal que se manexa para a visualización deste tipo de

información son más que suficientes para realizar as probas.

- Un conxunto de episodios de isquemia detectados sobre os sinal dos parámetros do diagrama de tendencias.
- Para as probas de parámetros fisiológicos, dispúxose de dous meses de medicións diárias de peso e tensión arterial (sistólica e diastólica.)
- Varios fragmentos de electrocardiograma con 3 canles diferentes de entre 1 e 5 minutos, e outro de media hora de duración. Isto é importante dado que nos requisitos exponse que a área de representación de electrocardiograma debe permitir a visualización de fragmentos de ata 5 minutos de duración. Ademais disto, replicouse a información dos 3 canais de electrocardiograma disponibles para poder probar a visualización coas 12 canles que se poden chegar a obter.
- Un conxunto de familias morfolóxicas detectadas sobre o electrocardiograma, cada unha delas con dúas canles diferentes.

6.2. Deseño das probas

Nesta sección expoñeremos o conxunto de probas que cada requisito debe superar para poder consideralo válido. Definirase para cada requisito unha batería de probas que se deberán realizar ó longo do desenvolvemento, e unha vez finalizado este, para garantir a validez do sistema. As probas definiranse segundo o seguinte esquema:

- *ID*: código co formato $Px.y$ que identifica univocamente a cada proba, onde x indica o número do requisito, e y é o número que identifica a esa proba dentro do conxunto de probas deseñadas para un mesmo requisito.
- *Descripción*: Conxunto de accións que se deben realizar para levar a cabo a proba
- *Resultado*: Saída esperada do sistema unha vez realizado o conxunto de pasos anteriores.

Segundo este esquema, preséntase a continuación o conxunto de probas deseñadas:

Proba P.1

Requisito FN.1: Alta de usuarios

<i>ID</i>	<i>Descripción</i>	<i>Resultado</i>
P1.1	Acceder ó sistema como administrador → Acceder á área de xestión de usuarios → Seleccionar <i>Novo usuario</i> → Introducir os datos solicitados → Seleccionar <i>Crear usuario</i> .	O sistema da de alta o usuario e aparece na lista de usuarios.
P1.2	Acceder ó sistema como administrador → Acceder á área de xestión de usuarios → Seleccionar <i>Novo usuario</i> → Introducir os datos solicitados → Seleccionar <i>Crear usuario</i> → Saír do sistema → Iniciar de novo o sistema → Introducir os datos de acceso do usuario creado.	O sistema dá de alta o usuario e permite o acceso.

Proba P.2

Requisito FN.2: Baixa de usuarios

<i>ID</i>	<i>Descripción</i>	<i>Resultado</i>
P2.1	Acceder ó sistema como administrador → Acceder á área de xestión de usuarios → Seleccionar un usuario da lista → Seleccionar <i>Eliminar</i> → Confirmar a operación na ventá que se mostra.	O sistema dá de baixa o usuario e non lle permite o acceso.

Proba P.3

Requisito FN.3: Modificación de usuarios

<i>ID</i>	<i>Descripción</i>	<i>Resultado</i>
P2.1	Acceder ó sistema como administrador → Acceder á área de xestión de usuarios → Seleccionar un usuario da lista → Editar os seus datos → Pulsar <i>Gardar</i> .	O sistema modifica os datos do usuario e os novos datos aparecen na lista de usuarios.

Proba P.4

Requisito FN.4: Identificación de usuarios

ID	Descripción	Resultado
P4.1	Iniciar a aplicación → Introducir un usuario e contrasinal válidos.	A aplicación permite o acceso e mostra o escritorio.
P4.2	Iniciar a aplicación → Introducir un usuario e contrasinal incorrectos.	A aplicación non permite o acceso e mostra unha mensaxe de erro.

Proba P.6

Requisito FN.6: Alta de pacientes

ID	Descripción	Resultado
P6.1	Acceder á ventá de xestión de pacientes → Pulsar en <i>Engadir Paciente</i> → Encher o formulario que se mostra → Pulsar <i>Dar de alta</i> .	O paciente dáse de alta e aparece agora na lista de pacientes

Proba P.7

Requisito FN.7: Baixa de pacientes

ID	Descripción	Resultado
P7.1	Acceder á ventá de xestión de pacientes → Pulsar na acción <i>Eliminar</i> situada xunto ó seu nome na lista → Confirmar a operación na ventá que se mostra.	Se existen datos de seguimiento móstrase unha mensaxe de erro. Noutro caso o paciente elimínase do sistema e desaparece da lista.

Proba P.8

Requisito FN.8: Edición de perfís de pacientes

ID	Descripción	Resultado
P8.1	Acceder á ventá de xestión de pacientes → Seleccionar un paciente da lista → Ir á pestana <i>Perfil</i> → Modificar os seus datos persoais → Seleccionar o botón <i>Gardar</i> .	Os datos do paciente modifícanse, aparecendo os novos no resumo da lista de pacientes e no seu perfil.

Proba P.9

Requisito FN.9: Consulta de pacientes

<i>ID</i>	<i>Descripción</i>	<i>Resultado</i>
P9.1	Acceder á ventá de xestión de pacientes.	Móstrase unha lista cos pacientes dados de alta no sistema
P9.2	Acceder á ventá de xestión de pacientes → Introducir no campo de busca parte do nome dun paciente.	Móstrase unha lista con aqueles pacientes dados de alta no sistema que amosan algúun tipo de coincidencia co nome introducido no campo de busca.

Proba P.10

Requisito FN.10: Asignación de médicos a pacientes

<i>ID</i>	<i>Descripción</i>	<i>Resultado</i>
P10.1	Acceder á ventá de xestión de pacientes → Seleccionar un paciente da lista → Ir á pestana <i>Perfil</i> → Pulsar na frecha situada á dereita do despregable de selección de médicos asignados → Na lista de médicos que se mostra, seleccionar aqueles que se desexan asignar ó paciente, e de ser o caso, desmarcar aqueles que se considere preciso → Pulsar o botón <i>Gardar</i> .	Actualízase a lista de médicos do paciente.

Proba P.11

Requisito FN.11: Edición de etapas de seguimiento

ID	Descripción	Resultado
P11.1	Acceder á ventá de xestión de pacientes → Seleccionar un paciente da lista → Ir á pestana <i>Etapas de seguimento</i> → Seleccionar <i>Nova etapa de seguimento</i> → Introducir as datas de inicio e fin e unha descripción → Pulsar en <i>Crear etapa</i> .	Móstrase a etapa creada xunto ó resto de etapas de seguimento do paciente.
P11.2	Acceder á ventá de xestión de pacientes → Seleccionar un paciente da lista → Ir á pestana <i>Etapas de seguimento</i> → Seleccionar unha das etapas de seguimento da lista → Seleccionar <i>Eliminar etapa</i> . Pulsar en <i>Crear etapa</i> .	Se existen datos de seguimento asociados á etapa móstrase unha mensaxe de erro. Noutro caso a etapa desaparece da lista.

Proba P.12

Requisito FN.12: Consulta retrospectiva de pacientes

ID	Descripción	Resultado
P12.1	Acceder á ventá de xestión de pacientes → Seleccionar un paciente que non estea actualmente baixo seguimento → Seleccionar unha das etapas de seguimento que se lle teñan realizado → Pulsar o botón de visualización do seguimento.	Visualízase o seguimento da etapa na ventá de seguimento.

Proba P.13

Requisito FN.13: Consulta en tempo real de pacientes

ID	Descripción	Resultado
P13.1	Acceder á ventá de xestión de pacientes → Seleccionar un paciente que está actualmente baixo seguimento → Seleccionar unha das etapas de seguimento en curso → Pulsar o botón de visualización do seguimento.	Visualízase o seguimento da etapa actual na ventá de seguimento e alértase mediante notificacións da ocorrencia de novos eventos.

Proba P.14

Requisito FN.14: Calendario

ID	Descripción	Resultado
P14.1	Seleccionar a icona do calendario na parte superior dereita da ventá de seguimento.	Móstrase o calendario da aplicación.
P14.2	Seleccionar a icona do calendario na parte superior dereita da ventá de seguimento → Seleccionar un día do calendario.	A liña temporal desprázase ata o día seleccionado.

Proba P.15

Requisito FN.15: Sinalización de eventos no calendario

ID	Descripción	Resultado
P15.1	Seleccionar a icona do calendario na parte superior dereita da ventá de seguimento.	Móstrase o calendario da aplicación cun conxunto de días sinalados.

Proba P.16

Requisito FN.16: Liña de representación temporal de eventos

ID	Descripción	Resultado
P16.1	Acceder ó seguimento dun paciente.	Visualízase unha liña temporal na que aparecen representados os evento e episodios que tiveron lugar durante un intervalo de tempo.
P16.2	Acceder ó seguimento dun paciente → Seleccionar a liña temporal e arrastrala.	A medida que se arrastra a liña temporal a ventá temporal que se visualiza avanza ou retrocede no tempo.
P16.3	Acceder ó seguimento dun paciente → Facer scroll co rato enriba da liña temporal.	Ao facer scroll a escala temporal varía, aumentando ou reducindo o nivel de detalle segundo desprazemos a roda do rato.

Proba P.17

Requisito FN.17: Visualización dos detalles dun evento

ID	Descripción	Resultado
P17.1	Acceder ó seguimento dun paciente → Seleccionar un dos eventos da liña temporal.	Móstrase un cadro coa descripción do evento e os seus detalles.

Proba P.18

Requisito FN.18: Visualización dos detalles dun episodio

ID	Descripción	Resultado
P18.1	Acceder ó seguimento dun paciente → Seleccionar un dos episodios da liña temporal.	Visualízase na área de parámetros o sinal asociado ao parámetro do diagrama de tendencias sobre o que se detectou o episodio seleccionado. O fragmento de sinal que se visualiza está centrado no instante no que tivo lugar o episodio.

Proba P.19

Requisito FN.19: Franxa de sincronización temporal

ID	Descripción	Resultado
P19.1	Acceder ó seguimento dun paciente → Engadir un parámetro do diagrama de tendencias á visualización.	Visualízase na liña temporal unha franxa que indica o intervalo que se mostra na área de parámetros.
P19.2	Desprazarse temporalmente na área de parámetros.	A franxa da liña temporal adáptase ó novo intervalo visualizado. En caso de chegar a franxa ó límite da liña temporal que se visualiza, a liña temporal desprázase para permitir visualizar o intervalo actual.

Proba P.20

Requisito FN.20: Filtro de eventos representados

ID	Descripción	Resultado
P20.1	Acceder ó seguimento dun paciente → Facer clic no botón <i>Filtrar</i> da liña temporal → Marcar ou desmarcar o tipo de eventos que se desexan visualizar.	Visualízanse unicamente aqueles eventos do tipo seleccionado.

Proba P.21

Requisito FN.21: Selección de parámetros

ID	Descripción	Resultado
P21.1	Acceder ó seguimento dun paciente → Facer dobre clic nun parámetro da árbore de parámetros.	Visualízase na área de parámetros o sinal asociado a ese parámetro.
P21.2	Acceder ó seguimento dun paciente → Seleccionar un parámetro e arrastralo á área de visualización.	Visualízase na área de parámetros o sinal asociado a ese parámetro.

Proba P.22

Requisito FN.22: Agrupación de parámetros

ID	Descripción	Resultado
P22.1	Acceder ó seguimento dun paciente → Facer dobre clic nun parámetro da árbore de parámetros → Facer clic no botón + para engadir un novo grupo → Facer dobre clic noutro parámetro.	Visualízase nunha pestana o primeiro parámetro seleccionado e noutra o segundo.
P22.2	Acceder ó seguimento dun paciente → Arrastrar varios parámetros á área de visualización → Facer clic no botón + para engadir un novo grupo → Arrastrar de novo varios parámetros á área de visualización.	Visualízase nunha pestana o primeiro conxunto de parámetros e na segunda o segundo.

Proba P.23

Requisito FN.23: Representación do diagrama de tendencias

ID	Descripción	Resultado
P23.1	Acceder ó seguimento dun paciente → Facer dobre clic nun parámetro da árbore de parámetros de ECG.	Visualízase na área de parámetros o sinal asociado a dito parámetro, ocupando este todo o espacio dispoñible.
P23.2	Acceder ó seguimento dun paciente → Arrastrar varios parámetros dende a árbore de ECG á área de visualización.	Visualízase o sinal de ámbolos dous parámetros, repartíndose o espacio entre eles.

Proba P.24

Requisito FN.24: Edición de anotacións

ID	Descripción	Resultado
P24.1	Acceder ó seguimento dun paciente → Facer dobre clic nun parámetro da árbore de parámetros → Facer clic no menú <i>Anotación</i> → Seleccionar a ferramenta <i>Anotación puntual</i> → Facer clic nalgún punto da gráfica → Introducir o texto da anotación → Pulsar <i>Aceptar</i> .	Aparece un globo de texto co comentario introducido que sinala o punto indicado sobre a gráfica.
P24.2	Acceder ó seguimento dun paciente → Facer dobre clic nun parámetro da árbore de parámetros → Facer clic no menú <i>Anotación</i> → Seleccionar a ferramenta <i>Anotación intervalo</i> → Facer clic nalgún punto da gráfica → Arrastrar ata outro punto da gráfica → Introducir o texto da anotación → Pulsar <i>Aceptar</i> .	Aparece un globo de texto co comentario introducido e unha chave que indica o intervalo seleccionado sobre a gráfica.
P24.3	Acceder ó seguimento dun paciente → Facer dobre clic nun parámetro da árbore de parámetros → Facer clic no menú <i>Anotación</i> → Seleccionar a ferramenta <i>Anotación intervalo</i> → Facer clic nalgún punto da gráfica → Arrastrar ata outro punto da gráfica → Introducir o texto da anotación → Pulsar <i>Aceptar</i> → Pulsar sobre a anotación.	A anotación elimínase da gráfica.

Proba P.25

Requisito FN.25: Mostrar/ocultar anotacións

ID	Descripción	Resultado
P25.1	Acceder ó seguimento dun paciente → Facer dobre clic nun parámetro da árbore de parámetros → Facer clic na ferramenta <i>Activar/Desactivar anotacións</i> .	As anotacións visibles pasan a estar ocultas.
P25.2	Acceder ó seguimento dun paciente → Facer dobre clic nun parámetro da árbore de parámetros → Facer clic na ferramenta <i>Activar/Desactivar anotacións</i> .	Móstranse de novo tódalas anotacións.

Proba P.26

Requisito FN.26: Representación de parámetros fisiológicos

ID	Descripción	Resultado
P26.1	Acceder ó seguimento dun paciente → Facer dobre clic nun parámetro da árbore de parámetros fisiológicos.	Visualízase na área de parámetros o sinal asociado a dito parámetro, ocupando este todo o espacio disponible.
P26.2	Acceder ó seguimento dun paciente → Arrastrar varios parámetros dende a árbore de parámetros fisiológicos ata a área de visualización.	Visualízase o sinal de ámbolos dous parámetros, repartíndose o espacio entre eles.

Proba P.27

Requisito FN.27: Desprazamento temporal na área de parámetros

ID	Descripción	Resultado
P27.1	Acceder ó seguimento dun paciente → Seleccionar varios parámetros do diagrama de tendencias → Pulsar nun dos botóns de desprazamento da parte inferior da vista de diagramas de tendencias.	A ventá temporal visualizada desprázase no tempo.
P27.2	Acceder ó seguimento dun paciente → Seleccionar varios parámetros fisiológicos → Pulsar nun dos botóns de desprazamento da parte inferior da vista de parámetros fisiológicos.	A ventá temporal visualizada desprázase no tempo.

Proba P.28

Requisito FN.28: Área de representación de electrocardiograma

ID	Descripción	Resultado
P28.1	Acceder ó seguimento dun paciente → Acceder á área de representación de ECG → Seleccionar unha canle facendo dobre clic nun dos nodos da árbore de selección de canles da parte esquerda.	Visualízase na área de representación de ECG o sinal asociado á canle seleccionada.
P28.2	Acceder ó seguimento dun paciente → Acceder á área de representación de ECG → Arrastrar varias canles á área de representación de ECG.	Visualízanse na área de representación de ECG os sinais asociados ó conxunto de canles seleccionadas.

Proba P.29

Requisito FN.29: Ventá de representación de familias morfolóxicas

ID	Descripción	Resultado
P29.1	Acceder ó seguimento dun paciente → Acceder á área de representación de ECG → Seleccionar o botón <i>Familias Morfolóxicas</i> .	Visualízanse o conxunto de latidos detectados no instante de tempo que se visualiza na ventá de ECG.
P29.2	Acceder ó seguimento dun paciente → Acceder á área de representación de ECG → Seleccionar o botón <i>Familias Morfolóxicas</i> → Pulsar un dos botóns de desprazamento da parte inferior.	Visualízanse o seguinte ou anterior conxunto de latidos detectados.

Proba P.30

Requisito FN.30: Sincronización temporal de ventás

ID	Descripción	Resultado
P30.1	Acceder ó seguimento dun paciente → Seleccionar varios parámetros → Desprazarse na área de parámetros.	A liña temporal avanza no tempo conxuntamente coa área de parámetros.
P30.2	Acceder ó seguimento dun paciente → Seleccionar varios parámetros → Desprazarse na área de parámetros ata cambiar de día.	A liña temporal avanza no tempo conxuntamente coa área de parámetros e o calendario cambia de día.
P30.3	Acceder ó seguimento dun paciente. Desprazarse pola liña temporal.	O calendario vai cambiando, indicando sempre a data que aparece no centro da liña temporal.

Proba P.31

Requisito FN.31: Visualización de cuadrículas

<i>ID</i>	<i>Descripción</i>	<i>Resultado</i>
P31.1	Acceder ó seguimento dun paciente → Seleccionar varios parámetros → Seleccionar unha cuadrícula no control da parte inferior esquerda da área de parámetros.	Sobre o sinal dos parámetros visualízase a cuadrícula seleccionada.
P31.2	Acceder ó seguimento dun paciente → Seleccionar varios parámetros → Seleccionar un facendo clic sobre o sinal → Cambiar a cuadrícula en amplitud no cadre da parte inferior dereita da área de parámetros.	Sobre o sinal do parámetro seleccionado visualízase a nova cuadrícula seleccionada.
P30.3	Acceder ó seguimento dun paciente → Acceder á ventá de representación de ECG → Seleccionar unha cuadrícula no control da parte inferior dereita da ventá de representación de ECG.	Sobre a área de representación de electrocardiograma visualízase a cuadrícula seleccionada.

Proba P.32

Requisito FN.32: Manipulación de escalas de amplitud

<i>ID</i>	<i>Descripción</i>	<i>Resultado</i>
P32.1	Acceder ó seguimento dun paciente → Seleccionar varios parámetros → Seleccionar un facendo clic sobre o sinal → Cambiar a escala de amplitud no cadre da parte inferior dereita da área de parámetros.	Visualízase o parámetro seleccionado na escala indicada.
P32.2	Acceder ó seguimento dun paciente → Acceder á ventá de representación de ECG → Seleccionar unha escala de amplitud no control da parte inferior esquerda da ventá de representación de ECG.	Visualízanse as canles de ECG na escala de amplitud indicada.

Proba P.33

Requisito FN.33: Manipulación de escalas temporais

ID	Descripción	Resultado
P33.1	Acceder ó seguimento dun paciente → Seleccionar varios parámetros → Cambiar a escala temporal no control da parte inferior esquerda da área de parámetros.	Visualízanse os parámetros seleccionados na escala temporal indicada.
P33.2	Acceder ó seguimento dun paciente → Acceder á ventá de representación de ECG → Seleccionar unha escala temporal no control da parte inferior esquerda da ventá de representación de ECG.	Visualízanse as canles de ECG na escala temporal indicada.

Proba P.34

Requisito FN.34: Realización de medicións

ID	Descripción	Resultado
P34.1	Acceder ó seguimento dun paciente → Seleccionar un parámetro → Seleccionar a ferramenta <i>medición</i> → Facer clic sobre un punto do sinal e arrastrar o cursor.	Mentres se arrastra o cursor indícase a medición realizada entre o punto seleccionado e o punto más próximo ó cursor, tanto en tempo como en amplitude.
P34.2	Acceder ó seguimento dun paciente → Seleccionar un parámetro → Seleccionar a ferramenta <i>medición</i> → Facer clic sobre un punto do sinal e arrastrar o cursor ata outro punto.	Móstrase a medición realizada entre o primeiro punto e o segundo, tanto en tempo como en amplitude.
P34.3	Acceder ó seguimento dun paciente → Acceder á ventá de representación de ECG → Activar o modo <i>medición</i> facendo clic no botón da parte inferior dereita.	Aparecen na pantalla dúas liñas que seguen o cursor do rato na área de representación de ECG.
P34.4	Acceder ó seguimento dun paciente → Acceder á ventá de representación de ECG → Activar o modo <i>medición</i> facendo clic no botón da parte inferior dereita → Seleccionar dous puntos sobre unha das canles visibles.	Móstrase na pantalla a medición realizada entre ámbolos dous puntos, tanto en tempo como en amplitude.

Proba P.35

Requisito FN.35: Área de representación da variabilidade de frecuencia cardiaca

ID	Descripción	Resultado
P35.1	Acceder ó seguimento dun paciente Acceder á área de representación de ECG → Seleccionar o botón <i>HRV</i> .	Ábrese unha ventá independente na que se mostra a información asociada a variabilidade da frecuencia cardiaca do paciente para o instante que está sendo visualizado.

6.3. Avaliación heurística

Nesta sección analizaremos a usabilidade do sistema en base as “10 regras heurísticas de usabilidade de Nielsen” comentadas no apartado 6.3. Dado que non foi posible polo momento a realización de probas de usabilidade con usuarios reais, comentaremos a continuación cada un dos puntos descritos por Nielsen amosando de que forma se tiveron en conta durante o deseño da ferramenta.

Visibilidade do estado do sistema

Este é un dos puntos no que se fixo maior fincapé durante o deseño. Nas diferentes ventás indícase sempre o intervalo que se está a visualizar, de forma textual e tamén en mediante o uso de franxes que o indican sobre a liña temporal. Tamén están visibles en todo momento as unidades da escala establecida, indicadores de todo tipo sobre que parámetros se visualizan, que canles están activadas, etc. Cando se reciben eventos novos móstranse notificación que o indican para que o usuario o saiba.

Utilizar a linguaaxe dos usuarios

En canto a isto, as mensaxes que se mostran utilizan a linguaaxe que habitualmente utiliza o persoal médico que vai traballar coa aplicación, ben de xeito explícito nas referencias e descripcións dos diferentes eventos, ou ben implicitamente, por exemplo, usando escalas e cuadrículas estándares que están habituados a utilizar.

Control e liberdade

Aínda que se trata de guiar ó usuario na interacción coa ferramenta, este é totalmente libre para navegar a través do seguimento realizado ao paciente, observando toda a información recollida.

Na área de visualización de parámetros, permítese organizar os diferentes parámetros en grupos segundo o usuario prefira, os cales serán visualizados concxuntamente, permitindo

engadir máis ou quitalos unha vez engadidos. Tamén é posible editar o nome dos grupos de xeito que o usuario os recoñeza máis facilmente.

Consistencia e estándares

Durante o deseño da aplicación, este punto tívose moi en conta, establecendo os mesmos mecanismos de interacción para procedementos similares sempre que foi posible. Por exemplo, para realizar a selección de parámetros ou de canles de electrocardiograma móstrase unha árbore dende a que estes se poden arrastrar á área na que van ser visualizados. Do mesmo xeito, para o desprazamento temporal utilizanse sempre o mesmo conxunto de controis.

Minimizar a carga da memoria do usuario

A arquitectura de información proposta trata de guiar ó usuario dende un nivel de pouco detalle da información ata un nivel moi detallado, tratando de que se centre naquela información relevante, e facendo énfase no uso de metáforas visuais facilmente asimilables, que lle permitan centrar os seus esforzos na comprensión da información e non da ferramenta. Por outra banda, o acceso ás ferramentas é sinxelo e atópase alí onde é necesario usalas, sen ter que recordar estruturas de menús ou outros controis mediante os que acceder a elas.

Flexibilidad e eficiencia de uso

A parte do xa comentado respecto á agrupación de parámetros, a ferramenta permite mostrar ou ocultar paneis segundo o usuario prefira, e configurar diferentes opcións. As iconas usadas tratan de ser autoexplicativas co fin de lograr unha maior eficiencia á hora de usalas, e os mecanismos de interacción adáptanse á natureza da acción para a que están deseñados (arrastar e soltar para engadir un parámetro á visualización, desprazar a liña temporal para avanzar ou retroceder no tempo, etc.). O espacio dispoñible é sempre utilizado completamente, por exemplo na representación de parámetros ou de electrocardiograma, independentemente do número de parámetros ou canles de ECG que se están a visualizar.

Por outra banda, xa comentou nos capítulos 4 e 5 os diferentes mecanismos utilizados para optimizar o rendemento da ferramenta como caches de datos locais que permiten un desprazamento polos sinais representados moi fluído, ou a carga dinámica de módulos que permite aforrar recursos.

Diálogos estéticos e deseño minimalista

Os mecanismos de notificación da aplicación son atractivos e non intrusivos, resaltan sobre o resto de componentes, e utilizan animacións sinxelas para aparecer e desaparecer, causando o mínimo impacto posible. Os diferentes botóns están normalmente indicados únicamente cunha icona que resulta simple identificar, e o texto móstrase cando o rato se pasa por enriba. Deste xeito pode suceder que a primeira vez o usuario necesite ler o texto, pero unha vez asociada a imaxe coa acción o texto é innecesario e aproveítase mellor o espazo dispoñible.

6.4. Validación

A continuación móstrase unha lisa cos diferentes requisitos do proxecto, xustificando o cumpriamento de cada un deles unha vez finalizado o desenvolvemento.

Requisito RD.1 Implementación en Java dos módulos de persistencia

Estado: Cumprido

Xustificación: O servidor está integralmente implementado en java, incluido o motor de persistencia.

Tests: Non procede.

Requisito RD.2 Utilización de librarías non privativas na medida do posible

Estado: Cumprido

Xustificación: Como se comentou no apartado 4.4.7, as librarías utilizadas son de carácter gratuito. As razóns que nos levaron a súa utilización e as diferentes comparativas realizadas exponenese nese mesmo apartado.

Tests: Non procede.

Requisito FN.1 Alta de usuarios

Estado: Cumprido

Xustificación: A aplicación permite que un administrador dea de alta usuarios a través da ventá de xestión de usuarios.

Tests: P.1

Requisito FN.2 Baixa de usuarios

Estado: Cumprido

Xustificación: A aplicación permite que un administrador dea de baixa usuarios a través da ventá de xestión de usuarios.

Tests: P.2

Requisito FN.3 Modificación de usuarios

Estado: Cumprido

Xustificación: A aplicación permite que un administrador edite o perfil dun usuario a través da ventá de xestión de usuarios.

Tests: P.3

Requisito FN.4 Identificación de usuarios

Estado: Cumprido

Xustificación: Durante o inicio do sistema móstrase unha ventá na que o usuario se debe identificar para poder acceder ó sistema. No servidor establecense mecanismos de validación que impiden que un usuario non identificado poida acceder a calquera información.

Tests: P.4

Requisito FN.5 Xestión de perfiles de usuario

Estado: Cumprido parcialmente

Xustificación: Un usuario pode xestionar como se mostran diferentes aspectos da interface, mostrar e ocultar paneis, organizar parámetros en grupos segundo prefira ou personalizar o número de familias morfolóxicas que se mostran en cada páxina. Porén estas opcións non persisten dunha sesión a outra.

Tests: Non procede.

Requisito FN.6 Alta de pacientes

Estado: Cumprido

Xustificación: Dende a ventá de xestión de pacientes é posible dar de alta no sistema novos pacientes.

Tests: P.6

Requisito FN.7 Baixa de pacientes

Estado: Cumprido

Xustificación: Dende a ventá de xestión de pacientes é posible dar de baixa pacientes do sistema.

Tests: P.7

Requisito FN.8 Edición de perfís de pacientes

Estado: Cumprido

Xustificación: Dende a ventá de xestión de pacientes é posible editar o perfil dos pacientes dados de alta no sistema.

Tests: P.8

Requisito FN.9 Consulta de pacientes

Estado: Cumprido

Xustificación: Dende a vista inicial da ventá de xestión de pacientes pódense consultar os pacientes dados de alta no sistema, podendo realizar buscas sobre os mesmos segundo diferentes criterios.

Tests: P.9

Requisito FN.10 Asignación de médicos a pacientes

Estado: Cumprido

Xustificación: Dende a ventá de xestión de pacientes é posible modificar a lista de médicos que un determinado paciente ten asignados, tanto no momento de dalo de alta como posteriormente dende o seu perfil.

Tests: P.10

Requisito FN.11 Edición de etapas de seguimiento

Estado: Cumprido

Xustificación: Dende a pestana *Etapas de seguimento* do perfil dun paciente, pódense xestionar as súas etapas de seguimento.

Tests: P.11

Requisito FN.12 Consulta retrospectiva de pacientes

Estado: Cumprido

Xustificación: É posible realizar unha consulta de carácter retrospectivo de tódolos pacientes dados de alta no sistema, é dicir, acceder á visualización do seguimento das diferentes etapas que se foron creando para monitorizalo, e que xa remataron.

Tests: P.12

Requisito FN.13 Consulta en tempo real de pacientes

Estado: Cumprido

Xustificación: A aplicación notifica ó persoal clínico mentres este está utilizando a ferramenta no caso de producírense novos eventos. Estes eventos son engadidos automaticamente á liña temporal e os seus detalles e posibles sinais asociados poden visualizarse como se doutro evento calquera se tratase.

Tests: P.13

Requisito FN.14 Calendario

Estado: Cumprido

Xustificación: A interface conta cun calendario que permite a navegación a través do seguimento dun paciente.

Tests: P.14

Requisito FN.15 Sinalización de eventos no calendario

Estado: Cumprido

Xustificación: No calendario da aplicación móstranse sinalados os días con eventos significativos.

Tests: P.15

Requisito FN.16 Liña de representación temporal de eventos

Estado: Cumprido

Xustificación: A aplicación conta cunha liña de representación temporal de eventos que permite obter un resumo dos eventos más significativos que tiveron lugar durante un longo período de tempo. A liña temporal permite que o usuario se desprase no tempo e que o faga a distintos niveis de granularidade temporal.

Tests: P.16

Requisito FN.17 Visualización dos detalles dun evento

Estado: Cumprido

Xustificación: Seleccionando un evento na liña temporal pódese amósarse todos os seus detalles nun cadro emergente.

Tests: P.17

Requisito FN.18 Visualización dos detalles dun episodio

Estado: Cumprido

Xustificación: A aplicación permite visualizar ó sinal asociado ó parámetro sobre o que se detectou un episodio concreto simplemente facen clic sobre el.

Tests: P.18

Requisito FN.19 Franxa de sincronización temporal

Estado: Cumprido

Xustificación: Sobre a liña temporal visualízase unha franxa que indica o intervalo de tempo que se está visualizando na área de representación de parámetros.

Tests: P.19

Requisito FN.20 Filtro de eventos representados

Estado: Cumprido

Xustificación: Os eventos que se mostran na liña de representación temporal poden ser filtrados segundo o seu tipo.

Tests: P.20

Requisito FN.21 Selección de parámetros

Estado: Cumprido

Xustificación: Na parte dereita da área de parámetros móstrase unha xerarquía de parámetros, que poden ser seleccionados arrastrándoo á área de visualización ou ben facendo clic sobre eles.

Tests: P.21

Requisito FN.22 Agrupación de parámetros

Estado: Cumprido

Xustificación: Na área de visualización de parámetros é posible crear diferentes grupos nos que se pode visualizar diferentes parámetros de xeito conxunto.

Tests: P.22

Requisito FN.23 Representación do diagrama de tendencias

Estado: Cumprido

Xustificación: A ferramenta permite a visualización dos diferentes parámetros do diagrama de tendencias. Seleccionando o menú de parámetros de ECG móstrase a vista deste tipo de parámetros, que permite a súa visualización utilizando sempre o máximo espazo dispoñible.

Tests: P.23

Requisito FN.24 Edición de anotacións

Estado: Cumprido

Xustificación: A través do menú de utilidades que se mostra sobre un parámetro ó pasar o cursor sobre el, é posible a realización de diferentes tipos de anotacións sobre o sinal asociado a el.

Tests: P.24

Requisito FN.25 Mostrar/ocultar anotacións

Estado: Cumprido

Xustificación: A través do botón *Activar/Desactivar anotacións* que se mostra no menú de

utilidades de cada parámetro é posible activar ou desactivar a visualización das anotacións realizadas sobre cada parámetro.

Tests: P.25

Requisito FN.26 Representación de parámetros fisiológicos

Estado: Cumprido

Xustificación: A ferramenta permite a visualización de parámetros fisiológicos. Seleccionando o menú de parámetros fisiológicos móstrase a vista deste tipo de parámetros, que permite a súa visualización.

Tests: P.26

Requisito FN.27 Desprazamento temporal na área de parámetros

Estado: Cumprido

Xustificación: Mediante os controis de desprazamento situados na parte inferior da área de representación de parámetros é posible desprazarse temporalmente sobre o sinal dos diferentes parámetros visualizados, tanto para os do diarama de tendencias coma para os fisiológicos.

Tests: P.27

Requisito FN.28 Área de representación de electrocardiograma

Estado: Cumprido

Xustificación: A ventá de representación de ECG permite a visualización de sinais de electrocardiograma, podendo o usuario seleccionar a través de mecanismos sinxelos as canles que deseña ver en cada momento.

Tests: P.28

Requisito FN.29 Ventá de representación de familias morfológicas

Estado: Cumprido

Xustificación: A ventá de familias morfológicas permite a visualización das familias morfológicas detectadas sobre o electrocardiograma nun instante concreto, podendo o usuario acceder ó último fragmento de sinal sobre o que se detectou un latido concreto.

Tests: P.29

Requisito FN.30 Sincronización temporal de ventás

Estado: Cumprido

Xustificación: As diferentes áreas de representación de sinais e o calendario mantéñense sincronizadas temporalmente, de xeito que os desprazamentos temporais se realizan en todo o seu conxunto.

Tests: P.30

Requisito FN.31 Visualización de cuadrículas

Estado: Cumprido

Xustificación: Tanto na ventá de representación de electrocardiograma coma na área de representación de parámetros se permite o establecemento de diversas cuadrículas, ben estándar ou libres e configurables polo usuario, a través dos controis situados na parte inferior de cada unha das áreas. No caso da representación de parámetros permítese ademais o axuste de cuadrículas independentes para cada sinal representado.

Tests: P.31

Requisito FN.32 Manipulación de escalas de amplitud

Estado: Cumprido

Xustificación: Nas áreas de parámetros e de ECG permítese a configuración de diferentes escalas de amplitud, facilmente configurables a través dos controis da parte inferior de ámbalas dúas áreas. No caso da representación de parámetros permítese ademais o axuste de escalas de amplitud independentes para cada sinal representado.

Tests: P.32

Requisito FN.33 Manipulación de escalas temporais

Estado: Cumprido

Xustificación: Nos diferentes niveis de representación de información é posible analizar a información utilizando diferentes escalas temporais. Para o caso da representación de parámetros e de electrocardiograma permítense establecer escalas estándar que o persoal clínico está habituado a utilizar.

Tests: P.33

Requisito FN.34 Realización de medicións

Estado: Cumprido

Xustificación: Tanto sobre o sinal dos diferentes parámetros representados coma sobre o sinal de electrocardiograma é posible a realización de medicións de xeito moi sinxelo, utilizando as ferramentas de medición que están dispoñibles en cada área.

Tests: P.34

Requisito FN.35 Área de representación da variabilidade de frecuencia cardiaca

Estado: Non cumprido

Xustificación: A implementación desta característica foi proposta polo cliente nun dos últimos *sprints* da etapa de desenvolvemento, e foi engadida á lista de requisitos do proxecto como de prioridade *opcional*. Porén, a súa implementación podía poñer en perigo a finalización en prazo do proxecto e finalmente considerouse que o máis indicado era non levala a cabo.

Tests: P.35

Requisito CA.1 Alto rendemento

Estado: Cumprido

Xustificación: A aplicación demostra un alto rendemento no manexo do gran volume de información co que traballa. Os diferentes mecanismos de cacheado de datos implementados, as mostraxes realizadas sobre os datos para minimizar o conxunto de datos co que se traballa nas gráficas, ou a carga dinámica de módulos fan posible aforrar recursos ou demorar o máis posible a súa carga, permitindo que a ferramenta acade o rendemento esperado.

Requisito CA.2 Interface usable e ergonómica

Estado: Cumprido

Xustificación: A validación deste requisito foi presentada na avaliación heurística do apartado 6.3.

Tests: Como xa se comentou no apartado 6.1, ata o momento de redacción deste documento non foi posible a instalación da aplicación no entorno de producción, polo que non foi posible a realización de tests de usabilidade cos futuros usuarios reais da aplicación. Porén, o equipo de dirección do proxecto ten un elevado grao de experiencia nese ámbito e durante todo o proxecto tratáronse reiteradamente e con especial interese este tipo de cuestiós.

Requisito CA.3 Axuda

Estado: Cumprido

Xustificación: A aplicación conta cunha axuda que describe como realizar tódalas tarefas necesarias durante o manexo da aplicación.

Requisito CA.4 Internacionalización

Estado: Cumprido parcialmente

Xustificación: O deseño da ferramenta permite a súa internacionalización de xeito simple mediante a creación de *scripts* no que se redefinan os diferentes termos, mensaxes, etc. utilizados na interface para cada idioma. Porén ata o momento a ferramenta únicamente está traducida ó galego e non se permite cambiar o idioma.

Requisito AL.1 Automatización da persistencia

Estado: Cumprido

Xustificación: O motor de persistencia deseñado ofrece mecanismos de persistencia automáticos a partires do modelo de datos, permitindo estender o modelo de xeito simple e incorporar novos servizos se ter que realizar grandes cambios.

Tests: Este requisito foi validado mediante probas unitarias das diferentes clases que permiten o acceso á información, situadas nas clases de probas *SystemEventDaoTest*, *MeasureEventDaoTest*, *TestEventDaoTest*, *PatientDaoTest*, *MonitoringEpisodeDaoTest*, *ECGSignalsDaoTest*, *TherapeuticAdminEventDaoTest*, *SymptomEventDaoTest* e *PhysiologicalEventDaoTest*.

Capítulo 7

Conclusións

O obxectivo do presente Traballo Fin de Grao foi o deseño e implementación dunha interface para a visualización e xestión do seguimento domiciliario de pacientes, que permita o almacenamento e a representación de toda a información recollida a través da plataforma Servando. Este proxecto enmárcase no conxunto de actividades que integra o Proxecto de Investigación “AI-SENIOR: Razonamiento temporal y minería de datos en sistemas de monitorización ubicua para el cuidado de las enfermedades de EPOC y EC” (Ref. TIN2009-14372-C03-03).

Esta memoria recolle os aspectos más importantes das diferentes etapas do desenvolvemento do proxecto, poñendo de manifesto a satisfacción dos diferentes requisitos do proxecto e o cumprimento dos obxectivos expostos polo cliente, o que rematou coa obtención dun artefacto software completamente validado e funcional. Entre os aspectos más salientables da solución proposta están:

- Provese dun modelo de datos extensible e proporcionase un motor de persistencia que permite xestionar toda a información que se obtén a través dos dispositivos de seguimento de Servando de xeito eficiente, e minimizando a carga de programación necesaria para a inclusión de novos tipos de datos.
- Proporcionase unha interface que permite a visualización do seguimento domiciliario de pacientes, contextualizando os principais eventos de interese, permitindo acceder a toda a información relacionada de xeito sinxelo, e proporcionando unha serie de funcionalidades e ferramentas que permiten a súa análise, entre as que podemos destacar:
 - Visualización da información temporal relevante para o seguimento dun paciente mediante unha liña temporal que permite caracterizar de xeito rápido e sinxelo o ocorrido durante longos períodos de tempo, así como acceder á información máis detallada sobre a súa ocorrencia, mediante unha organización vertical do tempo en múltiples granularidades.
 - Visualización a longo prazo da evolución de parámetros fisiológicos.
 - Visualización agregada e personalizable de diagramas de tendencias.
 - Visualización de sinais de electrocardiograma, utilizando escalas e cuadrículas estándar.

- Posibilidade de realizar medicións sobre os sinais representados, facilitando a identificación de sucesos anómalos e axudando á tarefa de diagnóstico.
 - Posibilidade de realizar anotacións sobre os sinais representados, facilitando a caracterización, interpretación e comunicación entre o persoal clínico.
- Faise uso de metáforas visuais que facilitan a análise dos datos en diferentes granularidades temporais, e mecanismos de interacción intuitivos que permiten un manexo simple da ferramenta, nun contexto de uso caracterizado pola sobrecarga de información.
 - Posibíltase a extensión do sistema, ofrecendo estratexias de almacenamento e xestión de información automatizadas e unha interface de usuario modularizada e facilmente extensible.
 - Preséntase unha solución robusta a funcional que está lista para ser implementada nun entorno real de seguimento domiciliario de pacientes.
 - Toda a funcionalidade requirida foi desenvolvida na esixencia da máxima innovación, poñendo a proba e experimentando coas últimas tecnoloxías e ferramentas de desenvolvemento no eido das aplicacións web, e demostrando a súa viabilidade técnica.

Ademais das características básicas que aseguran o cumprimento dos obxectivos do proxecto, hai que destacar outros aspectos do desenvolvemento orientados a mellorar a calidade global do produto. Neste sentido, a solución presentada:

- Baséase en tecnoloxías abertas como HTML, CSS e Javascript para a construcción das componentes visuais da interface, ou JSON para a transferencia de datos entre o cliente e o servidor, que aseguran a portabilidade do sistema entre diferentes plataformas.
- Implementa mecanismos de optimización e mellora do rendemento como o almacenamento intermedio (*cache*) de datos ou o almacenamento de sinais en campos binarios, permitindo a súa xestión eficiente.

Debemos tamén poñer de manifesto a importancia da componente metodolóxica neste proxecto, que permitiu levalo a cabo de forma rigorosa e predictable. En canto á decisión de utilizar *Scrum* como metodoloxía de desenvolvemento, cremos que foi acertada, polas seguintes razóns:

- O conxunto de requisitos inicial incrementouse ó longo do proxecto, redefiníndose a maioría dos requisitos existentes, algúns deles incluso máis dunha vez.
- O feito de involucrar ao cliente de xeito continuado, e a realización dun número elevado de encontros, aumentou a motivación durante o desenvolvemento e mantivo un ritmo de traballo constante e alto.
- O resultado obtido alíñase coas expectativas do cliente, debido ó proceso continuo de revisión que permitiu adaptarse á evolución do proxecto.
- A maioría dos riscos existentes ó comezo do proxecto foron mitigados durante as primeiras fases do mesmo, gracias a realización dun desenvolvemento en base a *sprints*, que permitiu ir completando a funcionalidade a medida que se probaban certos aspectos que non estaban claros.

- O uso de *Acunote* foi positivo en moitos sentidos. A xestión de tarefas fíxose moi simple e o coñecemento do grao de avance en todo momento permitiu adecuar a carga de traballo. Hai que dicir tamén que algunhas das funcionalidades que ofrece, coma o análise de tarefas realizadas non foron usadas do mellor xeito posible, e poderían ternos proporcionado unha serie de beneficios que mellorarían aínda máis o desenvolvemento do proxecto.

Finalmente, expóñense a continuación as posibles liñas de extensión e mellora da interface:

- Visualización dos resultados de cuestionarios, xa sexa mediante a súa visualización na liña temporal ou mediante a implementación dalgún módulo específico.
- Establecemento dun sistema de xestión de perfís de usuario que permita configurar dinamicamente os componentes da interface segundo as súas preferencias.
- Incorporación dunha *bandeira de entrada* semellante á usada nos sistemas de xestión de correo electrónico, que permita ó persoal clínico saber se se produciron eventos relevantes dende o seu último acceso ó sistema.
- Adaptación da interface de usuario a dispositivos táctiles, que permitan manexar os diferentes componentes mediante este tipo de interacción.

Apéndice A

Manual de usuario

A.1. Instalación

A continuación explicarase os conxunto de pasos a seguir para poñer en funcionamento a ferramenta. As instrucións presentadas non son específicas para ningún sistema en concreto e permiten a instalación da ferramenta sobre calquera tipo de plataforma.

Para instalar a aplicación é necesario ter instalados previamente un servidor de aplicacíons e un servidor de bases de datos PostgreSQL. Por comodidade suporemos que ámbolos dous servidores están instalados e se están executando localmente, o primeiro no porto 8080 e o segundo no porto 5432.

En primeiro lugar debemos crear unha nova base de datos e outorgarlle privilexios sobre ela a un usuario calquera. Despois debemos engadir manualmente a táboa na que se almacenarán os diferentes sinais en forma de *blobs*.

Para creala executamos o seguinte conxunto de sentencias SQL:

```
CREATE SEQUENCE blob_reference_id_seq;
CREATE TABLE blobreference (
    id bigint DEFAULT nextval('blob_reference_id_seq') PRIMARY KEY,
    data oid
);
```

Unha vez feito isto, editamos o ficheiro *persistence.xml* que se atopa dentro do ficheiro de despregue da aplicación (*ServandoUI.war*), e cambiamos os parámetros de acceso á base de datos polos da que acabamos de crear, como se mostra a continuación:

```
hibernate.connection.url ⇒ jdbc:postgresql://localhost:5432/NOMEBD
hibernate.connection.username ⇒ USUARIO
hibernate.connection.password ⇒ CONTRASINAL
```

Unha vez establecidos os parámetros de conexión coa base de datos, debemos despregar a aplicación no servidor de aplicacíons. Para isto, utilizamos a páxina de xestión do servidor Tomcat para indicarle o ficheiro de despregue citando anteriormente. Podemos acceder a dita páxina mediante un navegador web, indicando a dirección `http://localhost:8080/manager/`.

Unha vez subido este ficheiro ó servidor será recoñecido automaticamente e despregarse a aplicación. Despois accedemos á dirección `http://localhost:8080/ServandoUI/exportDatabase`, para que se xeren o resto de táboas necesarias para o funcionamento da aplicación.

Feito isto, a aplicación queda totalmente lista para funcionar e pódese acceder a ela dende un navegador indicando a dirección `http://localhost:8080/ServandoUI/`.

A.2. Axuda

Nesta sección explícase o funcionamento da ferramenta. En primeiro lugar descríbese a realización das tarefas de administración de pacientes e usuarios, e dado que son tarefas simples, faise indicando a secuencia de operacíons que se deben levar a cabo para cada unha delas. De seguido explícanse o funcionamento da ferramenta para a visualización do seguimento, dun xeito máis explicativo, dado que son tarefas que requieren dunha interacción máis complexa co sistema.

A.2.1. Xestión de usuarios

Creación de usuarios

1. Acceder ó sistema como administrador.
2. No menú principal, acceder a *Xestión de usuarios*.
3. Seleccionar *Engadir usuario*.
4. Cumprimentar os datos do usuario e seleccionar o tipo de usuario.
5. Seleccionar *Crear usuario*.
6. O usuario queda dado de alta.

Edición de usuarios

1. Acceder ó sistema como administrador.
2. No menú principal, acceder a *Xestión de usuarios*.
3. Seleccionar na lista de usuarios o usuario que se deseja editar.
4. Modificar os datos necesarios.
5. Seleccionar *Gardar*.
6. Os cambios quedan gardados.

Eliminación de usuarios

1. Acceder ó sistema como administrador.

2. No menú principal, acceder a *Xestión de usuarios*.
3. Seleccionar na lista de usuarios o usuario que se desexa eliminar.
4. No perfil do usuario, seleccionar *Eliminar usuario*.
5. Seleccionar *Gardar*.
6. Os cambios quedan gardados.

A.2.2. Xestión de pacientes

Busca de pacientes

1. No menú principal, acceder a *Pacientes*.
2. No campo de busca indicar o nome ou apelidos do paciente buscado.

Alta de pacientes

1. No menú principal, acceder a *Pacientes*.
2. Seleccionar *Engadir paciente*.
3. Cumprimentar os datos do usuario e seleccionar da lista de médicos dispoñibles os médicos que se ocuparán do seu seguimento.
4. Seleccionar *Dar de alta*.
5. O paciente queda dado de alta.

Baixa de pacientes

Esta funcionalidade pódese levar a cabo únicamente con aqueles pacientes sobre os que non se realizara ningún tipo de seguimento. En caso contrario non se permitirá.

1. No menú principal, acceder a *Pacientes*.
2. Seleccionar da lista de pacientes o paciente que se desexa dar de baixa.
3. Seleccionar *Dar de baixa*.
4. No diálogo que se mostra, pulsar *Confirmar*.

Edición de pacientes

Edición do perfil dun paciente

1. No menú principal, acceder a *Pacientes*.
2. Seleccionar da lista de pacientes o paciente que se desexa editar.
3. Na vista do usuario, ir á pestana *Perfil*.
4. Modificar os datos necesarios e seleccionar *Gardar*.
5. Os cambios quedan gardados.

Creación de etapas de seguimiento

1. No menú principal, acceder a *Pacientes*.
2. Seleccionar da lista de pacientes o paciente que se desexa editar.

3. No perfil do usuario, ir á pestana *Etapas de seguimento*.
4. Seleccionar *Nova etapa de seguimento*
5. Nos detalles da etapa, indicar as data de inicio e fin e introducir unha descripción da etapa.
6. Seleccionar *Crear etapa*

Eliminación de etapas de seguimento

1. No menú principal, acceder a *Pacientes*.
2. Seleccionar da lista de pacientes o paciente que se desexa editar.
3. No perfil do usuario, ir á pestana *Etapas de seguimento*.
4. Seleccionar a etapa que se desexa eliminar.
5. Seleccionar *Eliminar etapa*
6. No diálogo que se mostra, pulsar *Confirmar*.

Acceder á visualización do seguimento dun paciente

1. No menú principal, acceder a *Pacientes*.
2. Seleccionar o paciente na lista de pacientes.
3. Ir a pestana *Etapas de seguimento*.
4. Seleccionar unha das etapas de seguimento disponíveis
5. Seleccionar o botón circular que permite acceder ó seguimento.

A.2.3. Visualización do seguimento

A continuación descríbese a utilización da ferramenta para visualizar o seguimento de pacientes. O acceso ó seguimento dun paciente xa se detallou na sección anterior, polo que aquí se partirá de que se está visualizando a ventá de seguimento que se amosa na figura A.1

A.2.4. Visualización de eventos e episodios

A visualización dos eventos e episodios que tiveron lugar durante un período de tempo determinado é posible mediante a liña temporal de eventos (A.2).

A continuación explícase o seu funcionamento:

Visualizar os eventos dunha data concreta

Para ver os eventos dun día concreto na liña temporal non hai máis que seleccionar esa data no calendario. A liña temporal desprázase automaticamente a ese día.

Ver os detalles dun evento

Para visualizar os detalles dun evento hai que seleccionalo na liña temporal. Aparecerá un cadro emerxente que mostra as súas propiedades.

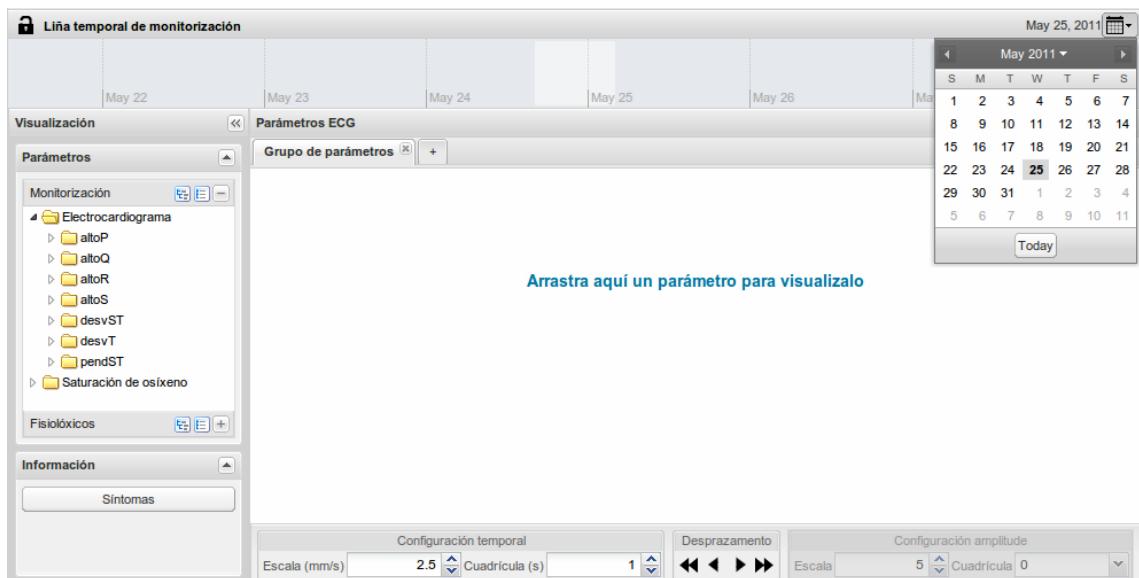


Figura A.1: Ventá de seguimento

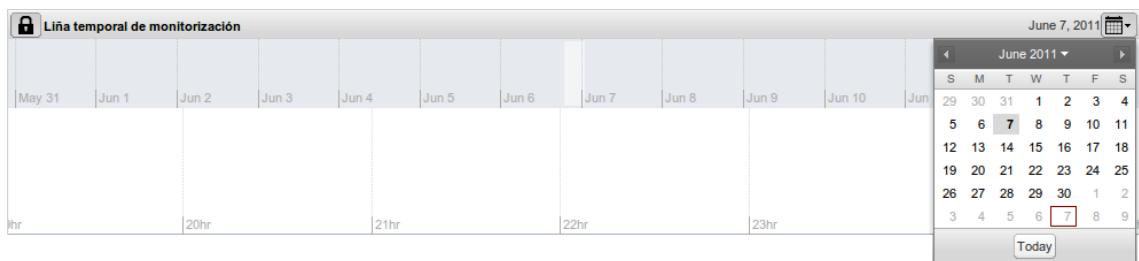


Figura A.2: Liña de representación temporal de eventos e episodios

Ver os detalles dun episodio

Para ver o sinal asociado ó parámetro sobre o que se detectou un certo episodio simplemente hai que facer clic na franxa que o representa na liña temporal. O sinal visualizarase na área de parámetros.

Desprazarse temporalmente

O desprazamento pola liña temporal pode realizarse dos seguintes xeitos:

- Arrastrar directamente a liña facendo clic co rato e desprazándoa no sentido que se deseche avanzar.
- Facer clic sobre a liña temporal e despois desprazarse usando as frechas do teclado.
- Seleccionar unha data no calendario, co que a liña se desprazará automaticamente a ela.

Cambiar o nivel de detalle

Para cambiar o nivel de detalle da liña temporal débese facer *scroll* sobre unha das súas bandas. A banda superior permite visualizar entre 1 e 15 días mentres que na inferior se poden ver unicamente entre 1 e 24 horas.

A.2.5. Visualización de parámetros

Vista de diagramas de tendencias

A vista de diagramas de tendencias permite visualizar os diferentes parámetros de ECG. Para seleccionar esta vista hai que facer clic no menú de parámetros de electrocardiograma do panel de ferramentas da dereita.

Vista de parámetros de fisiológicos

A vista de parámetros fisiológicos permite visualizar os diferentes parámetros fisiológicos durante intervalos de tempo de ata 15 días. Para seleccionar esta vista hai que facer clic no menú de parámetros fisiológicos do panel de ferramentas da dereita.

Selección de parámetros

Para seleccionar un parámetro e engadilo á visualización pódese facer dobre clic sobre el ou ben pódese pulsar sobre el e arrastralo ata a área de visualización.

Eliminación de parámetros

Para eliminar un parámetro da visualización débese seleccionar o botón *x* que se mostra na barra de ferramentas que aparece o pasar o rato por enriba do sinal.

Agrupamiento de parámetros

Para unha visualización máis cómoda pódense crear grupos de parámetros que se visualizan de xeito independente.

- Para **crear un grupo de parámetros** simplemente hai que facer clic no botón *+* situado na barra de pestanas.
- Para **eliminar un grupo de parámetros** da visualización hai que seleccionar o botón *x* situado na pestana correspondente ó grupo.
- Para **cambiar o nome dun grupo de parámetros** hai que facer dobre clic sobre o título do grupo, introducir o novo nome e pulsar *Enter* ou facer clic co rato noutro lugar calquera.

Cambiar escalas

Para cambiar a escala utilizada na área de representación de parámetros débese seleccionar a opción desexada da lista de escalas dispoñibles da parte inferior da dita área, ou axustar a escala de forma libre utilizando as frechas dese mesmo control. Para poder cambiar a escala de amplitud é necesario seleccionar antes un parámetro concreto facendo clic sobre el. Unha vez seleccionado o procedemento é o mesmo.

Cambiar cuadrículas

Para cambiar a cuadrícula utilizada na área de parámetros débese seleccionar a opción desexada da lista de cuadrículas dispoñibles da parte inferior de dita área. Para axustar a cuadrícula e necesario seleccionar antes un parámetro concreto facendo clic sobre el. Unha vez seleccionado o procedemento é o mesmo.

Desprazarse temporalmente

Para desprazarse temporalmente na ventá de parámetros débese seleccionar as frechas de

desprazamento da parte inferior («, <, >, »). As frechas exteriores permiten avanzar en intervalos iguais á ventá de tempo visible, mentres que as frechas interiores permiten o avance en intervalos da metade de tempo. O desprazamento pode conseguirse tamén seleccionando un parámetro calquera e pulsando despois as frechas de desprazamento do teclado.

Realizar medicións

Para realizar medicións débese facer clic no botón *medición* que aparece ó pasar o rato por enriba dun parámetro. Despois hai que seleccionar o punto de inicio da medición e arrastrar o rato ata o punto de fin da medición. O resultado da medición mostrárase na pantalla.

Realizar anotacións

Sobre o sinal dos diferentes parámetros poden realizarse anotacións de xeito sinxelo, mediante as ferramentas que se mostran o pasar o rato por enriba, e que se amosan na figura A.3.



Figura A.3: Ferramentas de edición de parámetros

- Para **anotar un instante** débese facer clic no botón *Anotación* que aparece ó pasar o rato por enriba dun parámetro. No menú que se mostra débese seleccionar a opción *Anotación puntual*. Despois chega con facer clic no punto que se desea anotar, e introducir o comentario no campo de texto que se mostra.
- Para anotar **anotar un intervalo** débese facer clic no botón *Anotación* que aparece ó pasar o rato por enriba dun parámetro. No menú que se mostra débese seleccionar a opción *Intervalo*. Para seleccionar o intervalo hai que seleccionar o punto de inicio do intervalo e arrastrar o rato ata o punto de fin do intervalo. Despois débese introducir o comentario no campo de texto que se mostra e confirmar a acción.
- Para activar ou desactivar as anotacións débese facer clic na icona con forma de ollo da barra de ferramentas.

A.2.6. Visualización de electrocardiograma

A ventá de representación de electrocardiograma (A.4) permite a representación deste tipo de sinais. A esta ventá accédese automaticamente facendo seleccionando as marcas de ECG que se mostran sobre os diferentes parámetros. A continuación móstrase as diferentes accións que é posible realizar utilizando esta ventá.

Seleccionar canles

Para seleccionar unha canle e engadila a visualización débese seleccionar esta na árbore de canles dispoñibles do panel de ferramentas da esquerda da ventá, ou ben seleccionala e arrastrala á zona de visualización.

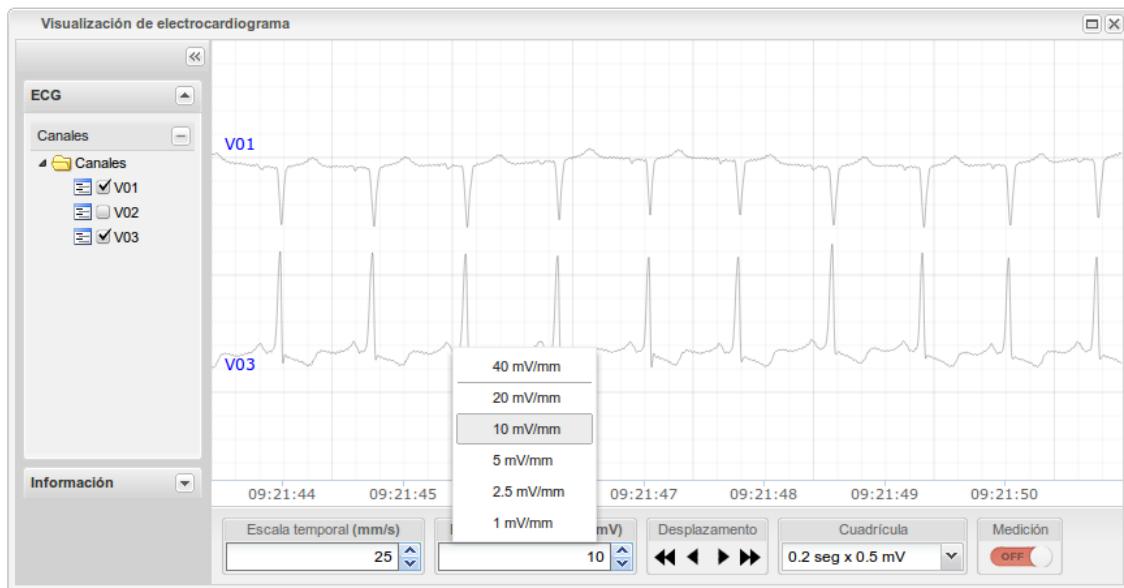


Figura A.4: Ventá de representación de electrocardiograma

Eliminar canles

Para eliminar unha das canles que se visualizan na área de visualización de electrocardiograma débese desmarcar a icona que representa esa canle na árbore de selección de canles.

Desprazarse temporalmente

Para desprazarse temporalmente na ventá de electrocardiograma débese seleccionar as frechas de desprazamento da parte inferior («, <, >, »). As frechas exteriores permiten avanzar en intervalos iguais á ventá de tempo visible, mentres que as frechas interiores permiten o avance en intervalos da metade de tempo.

Cambiar escalas

Para cambiar a escala utilizada na área de representación de ECG débese seleccionar a opción desexada da lista de escalas disponibles da parte inferior da dita área, ou axustar a escala de forma libre utilizando as frechas dese mesmo control.

Cambiar cuadrículas

Para cambiar a cuadrícula utilizada na área de representación de ECG débese seleccionar a opción desexada da lista de cuadrículas disponibles da parte inferior de dita área.

Realizar medicións

Para realizar medicións de tempo e amplitude sobre unha das canles que se visualizan na área de representación de ECG, é necesario activar o modo *Medición* facendo clic no botón da parte inferior dereita. Unha vez activado, móstranse dúas liñas que acompañan o rato e que axudan na medición. Para realizar a medición simplemente hai que seleccionar dous puntos sobre unha das canles e na pantalla mostraranse os resultados.

Ver familias morfolóxicas

Para acceder á visualización das familias morfolóxicas detectadas sobre o electrocardiograma, hai que facer clic no botón *Familias morfolóxicas* da parte inferior esquerda. Abrirase a ventá de representación de familias morfolóxicas (figura A.5) na que se poden realizar diferentes accións:

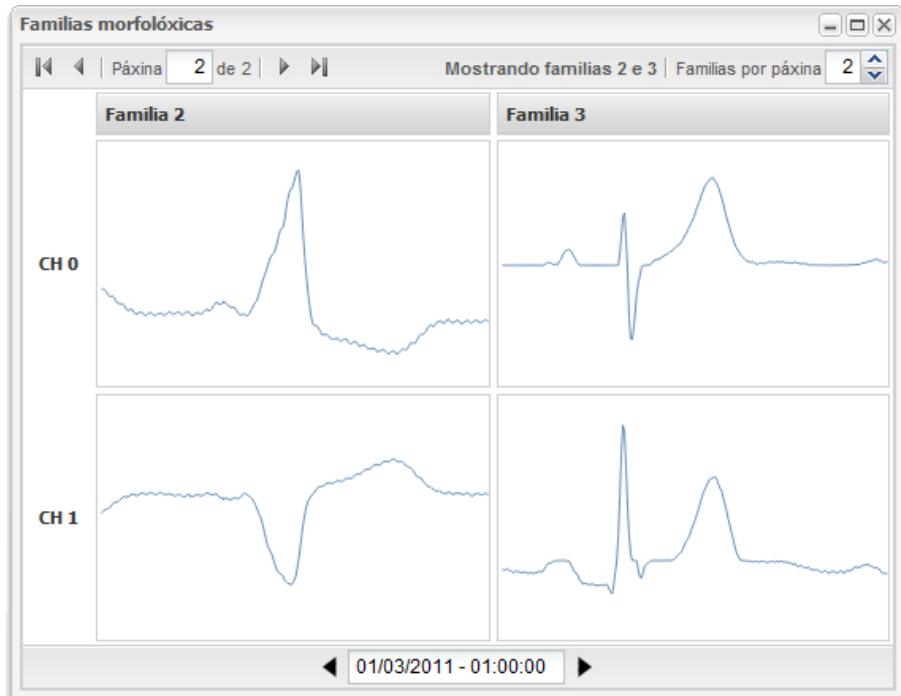


Figura A.5: Ventá de representación de familias morfolóxicas

- Para acceder o **seguinte conxunto de familias** hai que seleccionar o botón **>** da parte inferior.
- Para acceder o **anterior conxunto de familias** hai que seleccionar o botón **<** da parte inferior.
- Mediante o control da parte superior dereita pódese establecer o número de familias que se muestran en cada páxina, cando o número de familias é demasiado grande para velas todas xuntas. Unha vez establecido este valor, é posible desprazarse polas páxinas mediante os botóns da esquerda.

Bibliografía

- [1] <http://bugs.mysql.com/bug.php?id=1605>.
- [2] <http://dev.mysql.com/doc/refman/5.0/en/connector-j-reference-implementation-notes.html>.
- [3] <http://log4javascript.org/>.
- [4] Comparativa de implementaciones de jpa. <http://terrazadearavaca.blogspot.com/2008/12/comparativa-de-implementaciones-de-jpa.html>.
- [5] Dygraphs javascript visualization library. <http://dygraphs.com/>.
- [6] Jpa performance benchmark. <http://www.jpab.org/>.
- [7] ISO 9241-11. Guidance on Usability, 1998.
- [8] ISO/IEC FDIS 9126-1. Software Engineering-Product Quality-Part, 2000.
- [9] <http://www.ine.es/prensa/np620.pdf>, 2010.
- [10] Apache. Openjpa. <http://openjpa.apache.org/>.
- [11] L. Baresi, P. Fraternali, and G.J. Houben. *Web engineering: 7th international conference*. Springer, 2007.
- [12] Eclipse. Eclipselink. <http://www.eclipse.org/eclipselink/>.
- [13] R. Fairley. *Risk management for software projects*. IEEE Software, 1994.
- [14] M. Fowler. *UML distilled*. Addison-Wesley, 2004.
- [15] E. Gamma, R. Helm, R. Johnson, and John M. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Gamma, 1995.
- [16] Google. Google chart tools. <http://code.google.com/intl/es-ES/apis/chart/>.
- [17] A. Gándara and P. Félix. BEATLab, una herramienta para la visualización y caracterización del latido cardíaco. Traballo Fin de Grao, 2010.
- [18] Hibernate. Hibernate annotations. <http://www.hibernate.org/about/annotations>.

- [19] Hibernate. Relational Persistence for Java and .NET. <http://www.hibernate.org/>.
- [20] IEEE. Recommended practice for software requirements specifications. <http://ieeexplore.ieee.org/servlet/opac?pnumber=5841>, 1998.
- [21] M. Keith and M. Schincariol. *Pro EJB 3: Java persistence API*. Apress, 2006.
- [22] J. Linwood and D. Minter. *Pro Hibernate 3*. Apress, 2010.
- [23] J. Nielsen. *Usability engineering*. 2004.
- [24] Oracle. Oracle toplink. <http://www.oracle.com/technetwork/middleware/toplink/overview/index.html>.
- [25] J.A. Piñeiro, P. Félix, T. Teijeiro, and A. Gándara. Solicitud de aprobación de anteproyecto fin de grado. Documento interno, 2011.
- [26] K.H. Pries and J.M. Quigley. *Scrum Project Management*. CRC Press, 2010.
- [27] Sencha. Javascript framework for rich apps. <http://www.sencha.com/products/extjs/>.
- [28] B. Shneiderman. *Designing for fun: How to make user interfaces more fun*. ACM Interactions, 2004.
- [29] Highslide Software. Interactive javascript charts framework for rich apps. <http://www.highcharts.com>.
- [30] Ian Sommerville. *Ingeniería del software*. Pearson, 2005.
- [31] T. Teijeiro and P. Félix. Servando: Unha plataforma distribuída para a prestación domiciliaria de servizos médicos. Traballo Fin de Grao, 2010.
- [32] W3C. Document Object Model. <http://www.w3.org/DOM/>.
- [33] W3C. XML. <http://www.w3.org/XML/>.
- [34] W3C. XMLHttpRequest. <http://www.w3.org/TR/XMLHttpRequest/>.