MINISTERIO DE EDUCACIÓN SUPERIOR

UNIVERSIDAD DE HOLGUIN
OSCAR LUCERO MOYA

FACULTAD DE INFORMÁTICA Y MATEMÁTICA

Trabajo de Diploma para optar por el Título de Ingeniero Informático.

NeuroScreening.
Sistema Informático para la cuantificación de variables neurológicas en pacientes con Ataxia Espinocerebelosa tipo 2.

Autor: Michel Velázquez Mariño.

Tutores: DrC. Ing. Luis Cuevas Rodríguez.
DrC. Dr. Luis Velázquez Pérez.
Ing. Dinella Aguilera González.

Holguín 2011.

Dedicatoria:

Esta investigación está dedicada especialmente a las personas portadoras del gen de la Ataxia Espinocerebelosa tipo 2, sus familiares y amigos, espero que este contribuya con las investigaciones para mejorar su calidad de vida.

A mi familia, en especial a mis padres, abuelos y tíos más cercanos, por su apoyo, dedicación y guía constante.

A mi esposa, por las interminables noches de trabajo, su presencia constante y su cariño.

A mis suegros, por sus consejos para la vida.

Agradecimientos:

A mis tutores, el DrC. Luis Velázquez Pérez, el DrC. Luis Cuevas Rodríguez y la Ing. Dinella Aguilera González, por sus conocimientos aportados.

Al DrC. Rodolfo Valentín García Bermúdez y la MsC. Neysi León Pupo por sus exhaustivas revisiones y excelentes consejos.

Al personal científico del CIRAH y en especial al Lic. Roberto Rodríguez Labrada, por su dedicación.

A los muchachos de XYZtem y en especial a Roberto Becerra, por su ayuda y contribución.

A la Decana MsC. Carmen Pino Ávila, por aceptarme en la facultad.

A Aleida, por su apoyo incondicional.

A todos los que de una forma u otra ayudaron a la realización de este trabajo.

RESUMEN.

La Ataxia Espinocerebelosa tipo 2 es una enfermedad neurodegenerativa que se caracteriza por un síndrome cerebeloso asociado a trastornos de la motilidad ocular y de los reflejos osteotendinosos, en la provincia de Holguín las tasas de prevalencia e incidencia son las más altas a nivel mundial. Para valorar el grado de los trastornos estáticos y locomotores propios de los pacientes con esta afección se aplican diferentes pruebas, sin embargo, actualmente, no se cuenta con un equipo médico que sea capaz de aplicarlas de forma uniforme, integrarlas y permitir su configuración, así como la introducción de nuevos exámenes neurológicos, generación de diferentes reportes y seguimiento longitudinal de los pacientes para cuantificar la progresión de la enfermedad.

El Sistema Informático para la cuantificación de variables neurológicas en pacientes con Ataxia Espinocerebelosa tipo 2 propone la gestión de pacientes, el diseño y aplicación de estudios, así como la cuantificación y análisis de las variables neurológicas. Permite aplicar las pruebas Índice punto, Persecución Motora y Movimientos Alternados que demandan procesamiento y análisis de señales de movimiento, formación, evaluación y generación de gráficas de funciones matemáticas y análisis estadísticos de las variables involucradas. La solución fue desarrollada utilizando la metodología Proceso Unificado Ágil, con arquitectura basada en capas y programación orientada a objetos, sobre tecnologías libres.

El sistema se sometió a un conjunto de pruebas unitarias y de aceptación que arrojaron resultados satisfactorios, permitiendo afirmar que los métodos y herramientas seleccionadas para su desarrollo fueron acertados y que el sistema desarrollado es sencillo, organizado, fiable y preciso.

ABSTRACT.

The Spinocerebellar ataxia type 2 is a neurodegenerative disease characterized by a cerebellar syndrome associated to ocular motility disorders and tendon reflexes, in the province of Holguín the prevalence and incidence rates are the highest in the world. To assess the degree of static and locomotor disorders of the patients with this condition, the medical specialist applies several tests, however, currently does not exist a medical equipment that is capable of apply in an uniform way, integrate and customize these tests, besides the introduction of new neurological tests, the generation of several reports and longitudinal tracking of patients to quantify the progression of the disease.

The computer system for the quantification of neurological variables in patients with spinocerebellar ataxia type 2 suggests the management, design and implementation of studies and the quantification and analysis of neurological variables. It let apply the Index Point test, Persecution Motor test and Alternate Movements test, it demand signal processing and motion analysis, training, evaluation and generation of graphs of mathematical functions and statistical analysis of the variables involved. The solution was developed using the Agile Unified Process methodology, with architecture based on layers and object-oriented programming on free technologies.

The software was tested by a set of unit and acceptance tests, obtaining successful results, allowing to assert that the methods and tools selected for the development of it were accurate and that is simple, organized, reliable and accurate.

ÍNDICE.

ÍNDICE DE TABLAS.

ÍNDICE DE FIGURAS.

Introducción.

Las enfermedades heredo-degenerativas espinocerebelosas conforman un conjunto de entidades cuyo rasgo nosológico mejor definido en la actualidad es su carácter hereditario autosómico, dominante, recesivo o ligado al cromosoma X, aunque pueden observarse casos esporádicos (1).

Dentro de las enfermedades heredo-degenerativas se encuentran las ataxias hereditarias causadas por la degeneración del cerebelo y sus vías aferentes y eferentes, la médula espinal, los nervios periféricos y el tronco cerebral. Las Ataxias Espinocerebelosas (del inglés Spinocerebellar Ataxias, SCAs) son las más comunes dentro de estas ataxias (1,2).

Es preciso aclarar que el término ataxia, no define a una enfermedad específica ni a un diagnóstico determinado, sino al síntoma resultante del estado patológico de la coordinación de los movimientos. Con frecuencia, esta palabra se emplea para describir los trastornos de la marcha, los que se caracterizan por inestabilidad, descoordinación y aumento de la base de sustentación (3).

La Ataxia Espinocerebelosa tipo 2 (SCA2) es un subtipo de las SCAs, es provocada por la expansión de la repetición del trinucléotido CAG en la región codificante del gen ATXN2, siendo una de las formas moleculares con mayor tasa de prevalencia a nivel mundial, supera solo por la tipo 3 (1,4).

En Cuba, específicamente en la provincia de Holguín es la más frecuente de las SCAs. Las tasas de incidencia y prevalencia en esta zona ocupan el primer lugar a nivel mundial (1,5). Clínicamente se caracteriza por un síndrome cerebeloso asociado a trastornos de la motilidad ocular y de los reflejos osteotendinosos (6).

El síntoma inicial más frecuente es la dificultad en la marcha en el 94.28% de los enfermos, mientras que el 5.72% es la dificultad para hablar y la descoordinación de los movimientos en los miembros superiores e inferiores (1).

En los trastornos estáticos y locomotores propios de los pacientes con afecciones

cerebelosas se añade la dismetría, que se refiere a un trastorno de la coordinación de los movimientos voluntarios, que se caracterizan por ser anormalmente exagerados y desmesurados. Este signo se presenta en un alto por ciento de los pacientes cubanos con SCA2, y de ahí la importancia de su cuantificación (1,7).

En el año 2000 se creó el Centro para la Investigación y Rehabilitación de las Ataxias Hereditarias (CIRAH), iniciándose una nueva etapa en las investigaciones y asistencia de los enfermos y descendientes con riesgos de SCA2 y otras ataxias, las que por su rareza y poca incidencia a nivel mundial integran el grupo de las llamadas enfermedades huérfanas –que carecen de tratamientos–, esto, unido a la alta tasa de prevalencia en nuestro país representa un serio problema de salud.

Para valorar el grado de los trastornos estáticos y locomotores se diseñaron estudios que incluyeron las pruebas:

- Exploración cuantitativa del signo de Romberg.
- Maniobra Índice-punto.
- Estudio cuantitativo de la Diadococinesia.

con las que se demostró la existencia de diferencias estadísticamente significativas entre el grupo de sujetos enfermos y el de los sujetos sanos (8,9).

Las pruebas Maniobra Índice-punto y Estudio cuantitativo de la Diadococinesia se realizaron con el Sistema Automatizado para la cuantificación de los principales trastornos de la coordinación de los movimientos en afecciones neurológicas, modelo ATAX-1 desarrollado por el CIRAH en colaboración con la Universidad de Camagüey (10).

Sin embargo, actualmente, no se cuenta con un equipo médico que sea capaz de aplicar de forma estandarizada y con un margen mínimo de error estos exámenes, integrarlos y permitir su configuración, así como la introducción de nuevas pruebas neurológicas, generación de diferentes reportes, seguimiento longitudinal de los pacientes para cuantificar la progresión de la enfermedad y portabilidad para realizar consultas a domicilio y estudios de campo.

Siendo de vital importancia para el colectivo de investigadores del CIRAH desarrollar nuevas pruebas y técnicas automatizadas de diseño, aplicación y evaluación de exámenes para neurología cuantitativa con los siguientes propósitos:

- Evaluación terapéutica.
- Formulación de escalas clínicas objetivas.
- Evaluación de la rehabilitación.
- Descripción de las etapas evolutivas de la enfermedad.
- Desarrollo de modelos pronósticos.
- Correlación clínico-cuantitativa con variables moleculares (1,8).

Lo aspectos anteriormente expuestos, constituyen *la problemática* de este trabajo de investigación.

Problema Científico.

¿Cómo garantizar la integración, configuración, aplicación uniforme y evaluación de los exámenes para valorar la coordinación visuomotora en pacientes con SCA2?

Objeto de estudio.

La Neurología cuantitativa en las Ataxias Espinocerebelosas.

Campo de acción.

La aplicación de exámenes para cuantificar la coordinación visuomotora en pacientes con trastornos del movimiento.

Objetivo general.

Desarrollar un Sistema Informático que permita la integración, configuración, aplicación uniforme y evaluación de los diferentes exámenes para evaluar la coordinación visuomotora en pacientes con SCA2.

Objetivos específicos

- Realizar estudio del estado del arte de la aplicación de exámenes neurológicos para la evaluación de la coordinación visuomotora.
- Desarrollar la solución informática.
- Validar la calidad del producto.

Hipótesis.

Si se desarrolla un Sistema Informático que permita integrar, configurar, aplicar de forma uniforme y evaluar los exámenes para cuantificar la coordinación visuomotora en pacientes con SCA2, se podrá contribuir a la aplicación de estos exámenes de forma sencilla, organizada, fiable y precisa.

Tareas de la investigación.

1. Estudiar el diseño y aplicación de los exámenes de neurología cuantitativa para evaluar la coordinación visuomotora.
2. Analizar sistemas informáticos que se utilicen en la aplicación de pruebas de neurología cuantitativa.
3. Seleccionar las tecnologías óptimas para desarrollar la solución informática.

4. Realizar el análisis de la solución informática.
5. Realizar la valoración de sostenibilidad de la solución informática.
6. Realizar el diseño e implementación de la solución informática.
7. Someter a pruebas la solución informática.

Métodos Científicos de la Investigación (11,12)

- Métodos teóricos

  → *Método histórico-lógico:* Para comprender el estudio de la cuantificación de la coordinación, las investigaciones y los resultados referentes a esta área, la trayectoria de los estudios realizados a pacientes y controles en el CIRAH, condiciones bajo las cuales se realizaron y el grado de certeza de los mismos. Así como la comprensión de las leyes básicas del funcionamiento de los estudios y su principal esencia.

  → *Método sistémico:* Para detectar los subsistemas que componen el objeto, las relaciones entre estos y el flujo de ejecución de los procesos.

  → *Método de modelado:* Para representar los procesos implicados en la captura de las variables que intervienen en la cuantificación y el análisis automático de los trastornos de la coordinación.

- Métodos empíricos

  → *Método de la observación:* Para identificar las principales técnicas de aplicación de los exámenes neurológicos de cuantificación de la coordinación que se aplican a los pacientes.

  → *Método experimental:* Para verificar las teorías planteadas, así como para la toma de decisiones en la selección de las herramientas y metodologías para el

desarrollo exitoso de la investigación. Además para corroborar la hipótesis.

- → *La entrevista:* Para recolectar información, comprender mejor el negocio y extraer los requisitos funcionales y no funcionales.

Capítulo 1: Marco Teórico: En este capítulo se detallan los conceptos fundamentales tratados en el objeto de estudio de esta investigación, la situación del estado del arte de los diferentes exámenes, herramientas y técnicas en general para evaluar los trastornos de la coordinación visuomotora. Además se fundamenta la selección de la metodología de desarrollo de software y herramientas para el desarrollo de la solución.

Capítulo 2: Descripción de la solución: En este capítulo se muestran los principales procesos del desarrollo del software, haciendo hincapié en los requisitos funcionales, casos de uso del sistema, arquitectura y descripción e implementación de los principales elementos desarrollados para la conformación de este.

Capítulo 3: Validación de la solución: En este capítulo se realiza un análisis de la calidad de los resultados de la investigación mediante diferentes pruebas, profundizando en los requisitos funcionales, algoritmos críticos de procesamiento de datos, navegabilidad y comportamiento durante la aplicación de los exámenes. Además se hace una valoración de la sostenibilidad del proyecto.

1

Capítulo

Capítulo 1: Marco teórico.

Las ataxias hereditarias son un grupo de enfermedades clínicas, patológicas y genéticamente heterogéneas, que se encuentran dentro de los desordenes neurodegenerativos causados por la degeneración del cerebelo y sus vías aferentes y eferentes, la médula espinal, los nervios periféricos y el tronco cerebral (13).

Dentro de este grupo, los diferentes subtipos de de las SCAs son los más comunes, siete de los cuales (SCA1, SCA2, SCA3, SCA6, SCA7, SCA17, DRPLA) son causados por la expansión de la repetición del trinucléotido CAG (citosina-adenina-guanina) en la región codificante del gen mutado (7,13).

La Ataxia Espinocerebelosa tipo 2 dentro de las SCAs ocupa el segundo lugar a nivel mundial según su tasa de prevalencia, superada solo por la tipo 3 (14). En nuestro país es la más frecuente de todas las SCAs (15).

Un estudio epidemiológico reciente de la población cubana reveló la existencia de 578 pacientes con SCA2, pertenecientes a 163 familias distribuidas en 11 de las 14 provincias (División político-administrativa -*Ley Número 1304 del 3 de julio de 1976*-) que representa el 87% de los pacientes con SCAs del país, otorgando una prevalencia nacional de 6.57 casos por cada 100'000 habitantes. De las familias afectadas se detectaron 7'173 individuos asintomáticos con riesgo relativo de padecer la enfermedad, de estos el 42.98% son descendientes directos de individuos afectados lo que eleva la prevalencia nacional de la mutación a aproximadamente 28.51 casos por cada 100'000 habitantes (Fig. 1). La mayor concentración de la mutación se encuentra en la provincia de Holguín con el 61.40% de individuos enfermos o asintomáticos con riesgo relativo a padecer la enfermedad respecto al total de las mutaciones de la SCA2 en Cuba y una tasa de prevalencia de 182.75 casos por cada 100'000 habitantes (13).

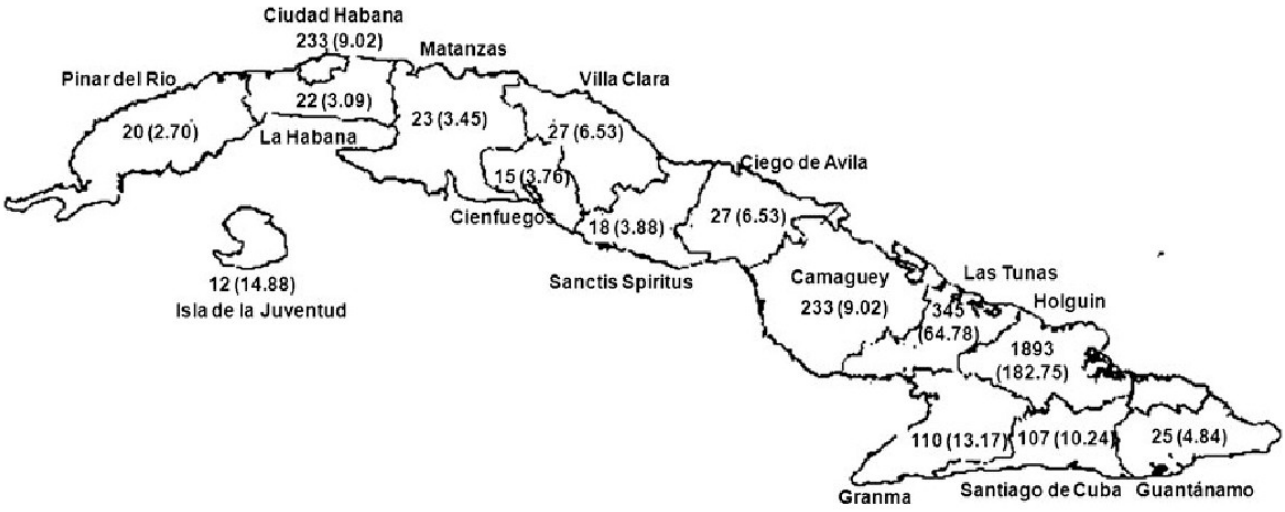

Fig. 1: Tasa de prevalencia de la mutación SCA2 en Cuba[1]*.*

Tomado de: A Comprehensive Review of Spinocerebellar Ataxia Type 2 in Cuba (13).

Dentro de las manifestaciones clínicas de la enfermedad se encuentran: ataxia de la marcha, dismetría, disartria cerebelosa, enlentecimiento en los movimientos oculares sacádicos, hipotonía, temblor cinético de miembros superiores e inferiores, contracturas musculares dolorosas, signo de Romberg, arreflexia osteotendinosa de miembros inferiores, hiporreflexia e hiperreflexia osteotendinosa y trastornos del sueño (13,16).

La edad de inicio de la enfermedad suele ubicarse en la cuarta década de vida, desde el cual esta continúa con un deterioro progresivo e implacable provocando la muerte del individuo entre los 10 y 15 años de evolución después de la aparición clínica de los síntomas (17).

1.1 Neurología cuantitativa.

La Neurología cuantitativa es un enfoque interdisciplinario que vincula a las neurociencias con las ciencias computacionales (18) con el objetivo de crear técnicas que sean capaces de cuantificar alteraciones neurológicas a través de diversos exámenes, establecer relaciones

[1] Los números fuera de los paréntesis representan el número absoluto de los portadores de la mutación (individuos con SCA2 más individuos con riesgo relativo), los números dentro de los paréntesis representan las tasas de prevalencias (portadores de la mutación SCA2 por cada 100'000 habitantes).

entre diferentes variables y profundizar en el conocimiento del funcionamiento del sistema nervioso.

1.1.1 Exámenes de neurología cuantitativa para evaluar la coordinación visuomotora.

La coordinación (taxia) es la combinación de contracciones de los músculos agonistas, antagonistas y sinérgicos, con el propósito de lograr movimientos voluntarios armónicos, coordinados y mesurados. Para que la ejecución de los movimientos sea coordinada es necesario integrar las funciones motoras y sensoriales. Durante la ejecución de cualquier movimiento y en especial durante los movimientos que involucran numerosos grupos musculares (movimientos complejos), es necesario que los músculos agonistas, antagonistas, sinérgicos y fijadores funcionen de manera coordinada y armónica (8,9,19).

La dismetría (hipermetría o hipometría) se refiere a un trastorno de la coordinación de los movimientos voluntarios que se caracterizan por ser anormalmente exagerados y desmesurados (1,20).

Aunque se han desarrollado diversos exámenes para cuantificar la función motora y sensorial, las pruebas para evaluar la ataxia no son comunes y en ocasiones su aplicación llega a ser compleja (21).

A continuación se relacionan un conjunto de pruebas utilizadas en la evaluación de los trastornos de la coordinación visuomotora.

Prueba Índice-punto, también conocida como dedo-nariz.

Básicamente consiste en ordenarle al sujeto que desde la posición de sentado, usando el dedo índice de su mano, toque de forma alternada la punta de su nariz y una diana que debe estar ubicada frente a él a una distancia máxima igual o inferior al alcance de su brazo. Esta prueba se aplica haciendo varias repeticiones del movimiento, con énfasis en la precisión y velocidad de ejecución. Existen diferentes variantes:

- Índice-nariz (del inglés Finger-to-nose test): Se ubica una pantalla transparente cuyo

centro coincida con la punta de la nariz del sujeto y se le ordena que extienda de forma horizontal el brazo con el que se está examinando hacia el lateral derecho o izquierdo –según el brazo en cuestión– y que acto seguido toque su nariz, este ejercicio se realiza de forma continua sin detener el movimiento, primero con los ojos abiertos y luego con los ojos cerrados. Se evalúa la distancia en centímetros entre el centro de la nariz y cada punto donde el sujeto tocó, por medio de las lineas concéntricas dibujadas en la pantalla (21).

- Índice-nariz estandarizado (del inglés Standardized Finger-Nose Test): Usando el dedo índice se debe tocar de forma alternada y tan rápido como sea posible en un periodo de 20 segundos la punta de la nariz y una diana horizontal ubicada a 45 centímetros de distancia frente al sujeto. La diana es un circulo rojo de 2 centímetros de diámetro que se puede desplazar en el eje vertical según la altura del sujeto. Si el sujeto no hace blanco de forma directa y precisa se le instruye que mueva su dedo hasta tocarlo antes de retornar el movimiento hacia su nariz. La precisión del movimiento no debe ser sacrificada por el aumento de la velocidad. En esta variante se registra el numero de repeticiones por cada ensayo (22).

Prueba Movimientos alternados (del inglés Tapping test).

Este examen es realizado con la ayuda de dos botones conectados a un sistema de conteo, situados uno al lado del otro a cierta distancia fija. El sujeto deberá estar en posición de reposo en una silla de frente al dispositivo y deberá tocar los botones izquierdo y derecho de forma alternativa con el dedo índice, tan rápido como sea posible. Cuando se realiza contra un período de tiempo se contabiliza el número de toques efectivos y cuando se realiza contra una cantidad de repeticiones se contabiliza el tiempo transcurrido entre cada toque y el tiempo total, en ambos casos se puede obtener el número de toques fallidos. En una segunda variante esta prueba puede aplicarse para evaluar la coordinación en los miembros inferiores (21).

Prueba Trazado de circulo (del inglés Circle-tracing task).

Para este examen se emplean dos pantallas, una colocada en posición horizontal y otra en posición vertical, estas muestran un circulo de 90 milímetros de diámetro con un anillo de color blanco de un grosor de 5 milímetros ubicado desde la frontera del circulo hacia adentro. El ejercicio consiste en trazar un circulo por el anillo de fondo blanco en la dirección de las manecillas del reloj, lo más rápido y preciso posible hasta completar una vuelta. Se puede realizar en dos variantes:

- Condición directa: Los sujetos pueden observar directamente el circulo y el camino trazado en la pantalla donde realizan el ejercicio.
- Condición indirecta: Los sujetos solo observan el circulo en la pantalla que se encuentra en posición vertical y no se muestra el camino trazado.

A través de esta prueba se evalúa la precisión y velocidad en el trazado, la transformación visuomotora y la detección y corrección de errores (23).

Prueba Cuantitativa de la Diadococinesia.

Se basa en la ejecución rápida de movimientos rítmicos alternados con las manos. El sujeto deberá estar en posición de reposo en una silla, desde esta posición se le solicita que golpee el muslo con su mano, alternando entre la palma y el dorso de esta. Este ejercicio se puede realizar con cada mano por separado o con las dos manos en conjunto de forma sincronizada[2] o alternada[3] (8,24).

Prueba Espiral de Arquímedes.

Este examen se puede realizar con la ayuda de algún sistema informático diseñado con este objetivo que capture los datos desde una pantalla táctil o simplemente con un lápiz y papel.

[2] Ambas manos alternan entre la palma y el dorso de forma simultanea.

[3] Ambas manos alternan entre la palma y el dorso de forma opuesta.

Para su ejecución el sujeto se coloca en posición de reposo en una silla frente al dispositivo de evaluación y se le instruye para que desde el centro del dispositivo dibuje una espiral de Arquímedes. Esta prueba se puede aplicar en dos variantes: dibujo libre de la espiral y dibujo guiado por una espiral de muestra. Permite el estudio de temblores y otras características cinemáticas de conductas en los brazos (25).

1.1.2 Sistemas Informáticos de neurología cuantitativa para evaluar la coordinación visuomotora.

BRAIN TEST©.

Es un programa de computadoras basado en el Sistema Operativo en Disco de Microsoft (del inglés Microsoft Disk Operating System, MS DOS), diseñado para computadoras personales IBM o compatibles. Es un software sencillo, rápido y fácil de usar, que permite cuantificar la función motora de los miembros superiores. Implementa el examen Tapping test donde el teclado es el dispositivo de prueba y las dos dianas son la tecla “S” y “;” que se encuentran a una separación de 15 centímetros en un teclado versión en inglés de 101/102 teclas. Las dos dianas son marcadas con un papel adhesivo de color rojo de 1 centímetro de diámetro. Los sujetos, para realizar el examen son sentados en una posición confortable de frente al teclado, a una altura que permita que sus brazos estén por encima del teclado cuando los codos se flexionan en un ángulo de 90 grados. La prueba se realiza por un periodo de 60 segundos tocando de forma alternada cada diana los más rápido y certero posible. Las variables que se analizan son: número de pulsaciones en 60 segundos, tiempo acumulado después que una tecla es liberada por más de 17 milisegundos, variación del intervalo de tiempo entre las pulsaciones e Índice ponderado, calculado utilizando el número de errores corregido para la velocidad (26,27).

Análisis Computarizado de la Espiral de Arquímedes.

Este es un método desarrollado con el lenguaje de programación C++ para el análisis del temblor y otras características cinéticas del comportamiento de los brazos derivados de la

escritura de la espiral de Arquímedes en una Tablet PC[4] a una resolución de 2540 puntos/pulgadas, con una precisión de ±0.005 pulgadas y 256 niveles de presión medible (2,5 g/nivel). Los datos de la espiral son recolectados en los ejes de coordenadas “X” y “Y”, y en el eje de la presión. En el análisis de los datos intervienen las variables: primer orden de suavidad, segundo orden de suavidad, presión, tasa de cruce y segundo orden de tasa de cruce; a partir de las variables más relevantes se calcula el grado de severidad. Los valores de las variables en conjunto con el grado de severidad proporcionan gran cantidad de información sobre el control motor de los brazos, que puede ser útil como un marcador inicial de afectaciones clínicas o servir como una medida objetiva de cambio después de intervenciones terapéuticas. Es una herramienta no invasiva, segura, rápida y portátil (28).

Sistema automatizado para la cuantificación de los principales trastornos de la coordinación de los movimientos en afecciones neurológicas. Modelo: ATAX-1.

Sistema informático desarrollado en el año 2000 por el CIRAH en colaboración con la Universidad de Camagüey. Permitía la aplicación de las pruebas punto-nariz y adiadococinesia. La prueba punto-nariz se ejecuta tomando como diana la letra “H” del teclado, tiene una duración de 40 segundos y se puede realizar en dos pasos (1 - ojos abiertos, 2 - ojos cerrados), en este ejercicio se contabiliza la cantidad de veces que fue tocada cada tecla y el tiempo entre un toque y otro. A partir de esos datos se calcula la efectividad general (por ciento de toques de la letra “H” respecto al total) y el período de los toques. La prueba de la adiadococinesia se realiza con la utilización de unos guantes diseñados específicamente con este objetivo, equipados con sensores ubicados en sus partes inferiores y posteriores. Tiene una duración de 1 minuto y a partir de la ejecución de esta se pueden calcular diferentes valores estadísticos de la frecuencia de los toques, la cantidad de toques y el valor integral[5]. El Sistema fue diseñado sobre tecnologías

[4] Computadora portable que no pose teclado, la entrada de instrucciones es a través de la pantalla táctil.

[5] $I = \sum_{i=1}^{NúmeroGolpes} \left(P_{Prom} - P_i\right)$

propietarias para los Sistemas Operativos Windows 95 y Windows 98 (10).

1.2 Métodos y herramientas.

Existen diversos métodos y herramientas para guiar y soportar el desarrollo de software, a continuación se analizan las principales características de las candidatas a ser seleccionadas.

1.2.1 Metodologías de desarrollo de software.

La construcción de un software es un proceso de aprendizaje iterativo y el resultado es el conjunto del software reunido, depurado y organizado mientras se desarrolla el proceso. Según Pressman el proceso del software es un marco de trabajo de las tareas que se requieren para construir software de alta calidad y define el enfoque que se toma cuando este es tratado por la ingeniería. Pero, la ingeniería del software también comprende los métodos técnicos y las herramientas automatizadas (tecnologías) que tiene el proceso (29).

La ingeniería de software es una tecnología multicapa donde el *proceso* define un marco de trabajo para un conjunto de las áreas claves, los *métodos* indican como construir técnicamente el software y las *herramientas* proporcionan un enfoque automático o semi-automático para el *proceso* y los *métodos* (29).

Para aplicar correctamente la ingeniería de software se utilizan diferentes metodologías de desarrollo.

Una metodología de desarrollo es un marco de trabajo que se emplea para estructurar, planificar y controlar el proceso de desarrollo de un sistema informático. Una gran variedad de marcos de trabajo han evolucionado a lo largo de los años, cada uno de ellos con fortalezas y debilidades (30).

Proceso Unificado de Rational (del inglés Rational Unified Process, RUP).

Es un Proceso de Ingeniería de Software que proporciona un enfoque disciplinado para la

asignación de tareas y responsabilidades dentro de una organización de desarrollo. Su objetivo es garantizar la producción de software de alta calidad, que satisfaga las necesidades de sus usuarios finales, dentro de un calendario y un presupuesto estimado (31).

Utiliza el Lenguaje Unificado de Modelado (del inglés Unified Modeling Language, UML), lenguaje estándar de la industria que permite comunicar con claridad requerimientos, arquitecturas y diseños (32).

El Proceso Unificado presenta tres características distintivas:

- Guiado por Casos de Uso: Utiliza los Casos de Uso para guiar el desarrollo desde la fase inicial hasta la fase final.
- Centrado en la Arquitectura: El proceso intenta comprender los aspectos estáticos y dinámicos más significantes en términos de arquitectura de software. La arquitectura se desarrolla en función de las necesidades de los usuarios, que son capturadas mediante los Casos de Uso.
- Iterativo e incremental: El proceso reconoce que es práctico dividir el proyecto en miniproyectos, donde la ejecución de cada miniproyecto representa una iteración que puede abarcar todos los flujos de trabajo del proceso y su resultado es un incremento. Las iteraciones son planificadas usando los Casos de Uso.

Los ciclos o iteraciones se repiten durante toda la vida del proyecto y cuentan con cuatro fases: Inicio, Elaboración, Construcción y Transición.

- Fase Inicio: Durante la fase inicial la idea central es desarrollada dentro de la visión del producto, se comprueba el entendimiento del negocio, se establece la viabilidad y el alcance del proyecto.
- Fase Elaboración: Durante esta fase la mayoría de los Casos de Uso son especificados en detalle y se diseña la arquitectura base del sistema, se identifican los riesgos más importantes, se prepara el cronograma y se estima el costo del proyecto completo.

- Fase Construcción: Durante la fase de construcción el producto se mueve de una arquitectura base a un sistema bastante completo para la transición hacia la comunidad de usuarios.
- Fase Transición: En esta fase el objetivo es asegurar que los requerimientos satisfagan a las partes interesadas, esta fase se inicia con frecuencia cuando se tiene una entrega *beta* de la aplicación. También incluye la terminación del manual y la identificación y corrección de errores.

Los flujos de trabajo que se realizan durante el proceso son: Modelado de negocio, Requisitos, Análisis y diseño, Implementación, Pruebas, Despliegue, Gestión del cambio y configuración, Gestión de proyecto y Entorno (Fig. 2) (33).

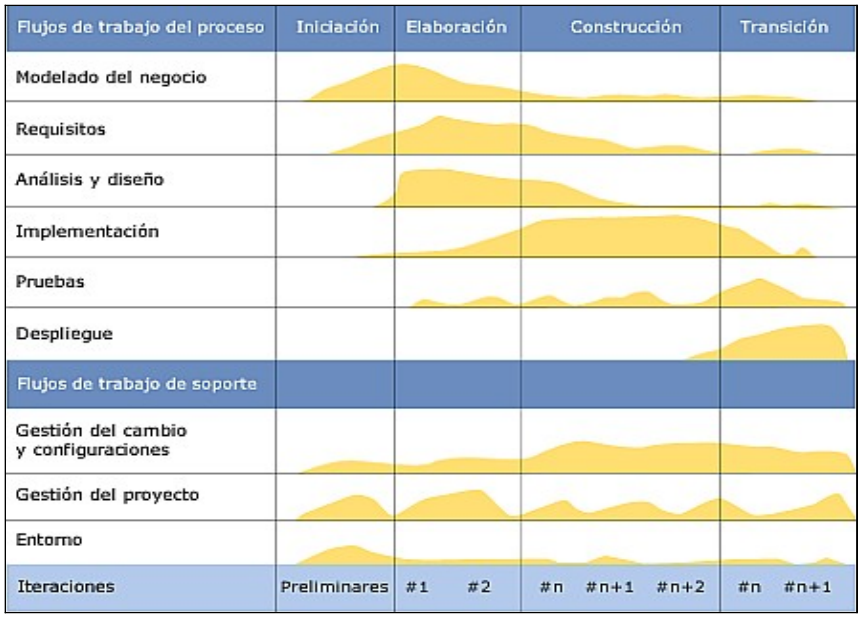

Fig. 2: Ciclo de vida de RUP.

Tomado de: http://es.wikipedia.org/wiki/Archivo:Rup_espanol.gif

Programación Extrema (del inglés eXtreme Programing, XP).

La Programación Extrema es una metodología de desarrollo de software ligera, se basa en la simplicidad, la comunicación y la realimentación o reutilización del código construido (34).

Está concebida para atender las necesidades específicas del proceso de desarrollo de software, llevado a cabo por equipos pequeños formados entre dos y diez programadores, que enfrentan requerimientos imprecisos y cambiantes. Esta metodología ligera desafía muchos principios convencionales, entre ellos el que plantea que el costo de cambiar una parte del software aumenta necesariamente de forma dramática a medida que se acerca al final de su desarrollo (35).

Se distingue de otras metodologías por sus principios concretos y la retroalimentación continua de los ciclos cortos; su enfoque en la planificación incremental; su capacidad para planificar de forma flexible la implementación de las funcionalidades, respondiendo a las necesidades cambiantes del negocio y su dependencia de la comunicación oral, las pruebas, y el código fuente para comunicar la estructura y la intención del sistema (35).

XP sigue un modelo de desarrollo desde la perspectiva de un sistema de control de variables, donde las variables implicadas son el costo, el tiempo, la calidad y el alcance. Cuenta, además, con cuatro fases de desarrollo: Planificación, Diseño, Desarrollo y Pruebas (Fig. 3) (34,35).

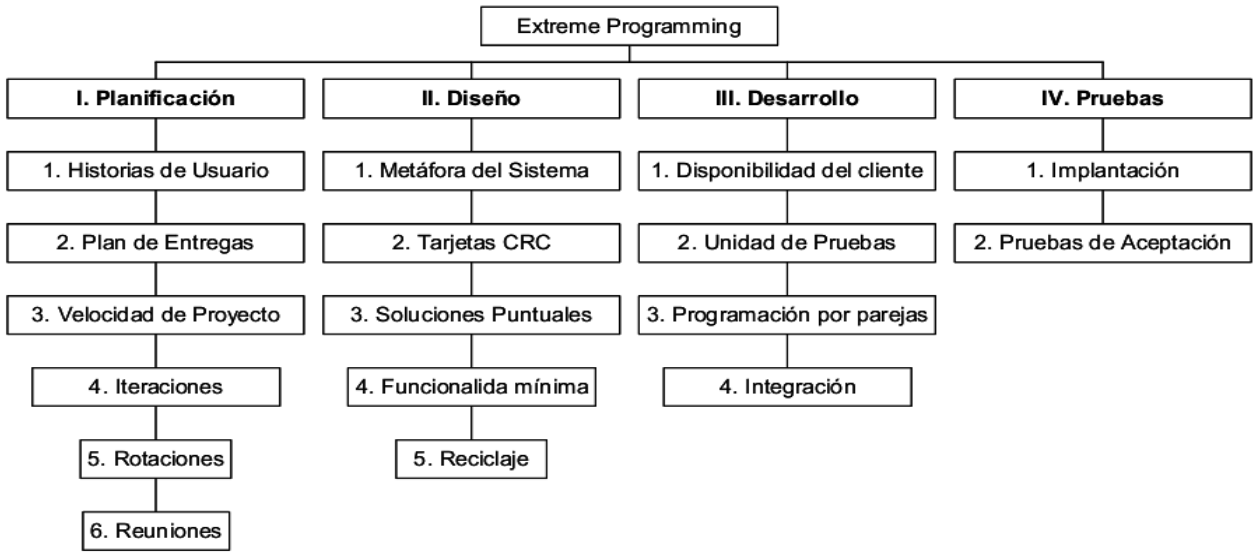

Fig. 3: Fases de la Metodología XP.

Tomado de: Extreme Programming Explained: Embrace Change (35).

Para un correcto desarrollo de XP se plantea que las versiones del software deben ser tan pequeñas como sea realizable, conteniendo los requisitos del negocio más importantes. El diseño debe contener el menor número de clases y métodos posibles, logrando que funcione con todas las pruebas, que no tenga lógica duplicada y que manifieste cada intención importante para los programadores. Cuando se desarrolla se debe hacer de la forma más sencilla posible sin perder en funcionalidad, la programación será por parejas de manera que mientras uno codifica el otro piensa la mejor forma de hacerlo. El cliente debe estar disponible todo el tiempo para el equipo de programadores, responder a sus preguntas, resolver discusiones y fijar las prioridades (36).

Proceso Unificado Ágil (del inglés Agile Unified Process, AUP).

AUP es un método ágil para el desarrollo de software basado en RUP, con las mismas prácticas y fases, pero con los flujos de trabajo simplificados y reducidos en número. Describe de forma sencilla un enfoque de desarrollo de software utilizando técnicas y conceptos ágiles. Se puede aplicar a una amplia gama de proyectos adaptándose a las necesidades de cada uno de ellos. Este enfoque aplica técnicas que incluyen Desarrollo Dirigido por Pruebas (del inglés Test Driven Development, TDD), Desarrollo Dirigido por Modelado Ágil (del inglés Agile Model Driven Development, AMDD), Administración Ágil de Cambios (del inglés Agile Change Management, ACM) y Recodificación de Bases de Datos (del inglés Database Refactoring, DR) que permiten ganar en productividad (37).

Otorga especial importancia a la gestión de riesgos. Propone que aquellos elementos con alto riesgo obtengan prioridad en el proceso de desarrollo y sean abordados en etapas tempranas del mismo. Para ello, se crean y mantienen listas identificando los riesgos desde etapas iniciales del proyecto. Especialmente relevante en este sentido es el desarrollo de prototipos ejecutables durante la base de elaboración del producto, donde se demuestra la validez de la arquitectura (38).

En AUP el Modelo abarca los flujos de trabajo de Modelado de Negocio, de Requerimientos y de Análisis y Diseño, además asume las actividades de Gestión de Cambios típicas en la parte de manejo de requisitos (37).

Al igual que en RUP, en AUP se establecen cuatro fases (Fig. 4) que transcurren consecutivamente.

- Fase Inicio: Durante esta fase el equipo de desarrollo en conjunto con el cliente definen el alcance del nuevo sistema y proponen una o varias arquitecturas candidatas para el mismo.
- Fase Elaboración: En esta fase el equipo de desarrollo profundiza en la comprensión de los requisitos del sistema y en validar la arquitectura.
- Fase Construcción: Durante la fase de construcción el sistema es desarrollado y probado en el ambiente de desarrollo.
- Fase Transición: El sistema se lleva a los entornos de pre-producción donde se somete a pruebas de validación y aceptación y finalmente se despliega en los sistemas de producción (37,38).

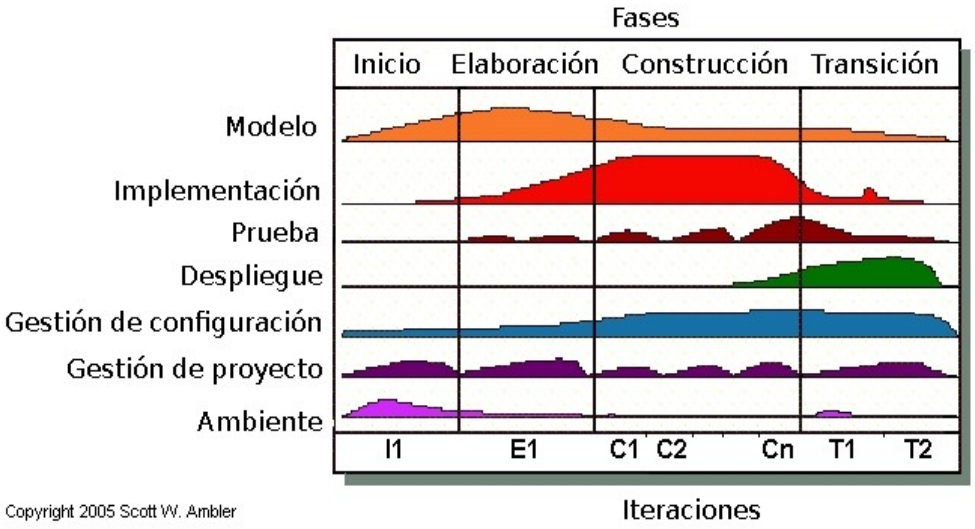

Fig. 4: Ciclo de vida de AUP.

Tomado de: The Agile Unified Process (AUP) Home Page [Internet] (37).

Las disciplinas se llevan a cabo de manera sistemática a fin de desarrollar, validar, y entregar el software de trabajo que responda a las necesidades de los clientes (38).

1.2.2 Lenguajes de programación.

El lenguaje en el que un programador de computadoras escribe instrucciones para que sean ejecutadas por esta se conoce como Lenguaje de Programación. Actualmente son usados cientos de estos lenguajes que se pueden clasificar en:

- Lenguajes de bajo nivel.
  - Lenguajes de máquinas.
  - Lenguajes de ensamblador.
- Lenguajes de alto nivel (39,40).

Los lenguajes de alto nivel permiten a los programadores escribir rutinas en un lenguaje asequible y utilizar notaciones matemáticas comunes. Estos lenguajes permitieron la creación de diversos paradigmas de programación, entre los más conocidos se encuentran la Programación Imperativa, la Programación Lógica, la Programación Funcional y la Programación Orientada a Objetos (POO) (41).

La POO permite resolver problemas basándose en un modelo organizativo de la vida cotidiana, donde los conceptos fundamentales se representan mediante objetos ampliamente autónomos, responsables de tareas específicas. Un objeto es una encapsulación del estado y del comportamiento de alguna clase (42).

Como características fundamentales presentes en la POO se encuentran la Abstracción, la Encapsulación de la información, la Herencia y el Polimorfismo.

Entre los lenguajes de programación orientados a objetos se destacan: C++, C#, Java, PHP

a partir de su versión 5, Ada, Object Pascal, Smalltalk, Visual Basic 6.0, VB.NET, Visual FoxPro en su versión 6 y Objective-C. Muchos de estos lenguajes son híbridos que combinan diferentes paradigmas de programación.

C++.

El lenguaje de programación C++ se comenzó a desarrollar en 1980 por Bjarne Stroustrup como extensión de C, su nombre hace referencia al operador incremento **++**. Es un lenguaje versátil, potente, rápido y de propósito general, mantiene las ventajas del C en cuanto a riqueza de operadores y expresiones, flexibilidad y eficiencia (43).

Incluye, nuevos tipos de datos, clases, herencia múltiple y polimorfismo, tipos de datos abstractos, plantillas, interfaces, mecanismos de manejo de excepciones, espacios de nombres, funciones *inline*, sobrecarga de operadores, referencias y operadores para manejo de memoria persistente (43).

Es reconocido por sus potencialidades para el desarrollo de aplicaciones robustas con requerimientos de recursos de hardware relativamente bajos, eficiencia en el uso de la Memoria de Acceso Aleatorio (del inglés Random Access Memory, RAM) y por brindar la posibilidad de utilizar indistintamente o a la vez la Programación Orientada a Objetos y la Programación Orientada a Algoritmos (programación procedural).

Su éxito entre los programadores profesionales le ha llevado a ocupar el primer puesto como herramienta de desarrollo de aplicaciones (44).

C#.

C# es un lenguaje de programación desarrollado por Microsoft que aunque está basado en C y C++, sus diseñadores decidieron no hacerlo compatible con estos para así evitar las debilidades de sus predecesores. Al diseñarlo se trató de mantener casi toda la sintaxis de las versiones C/C++ de manera que los programadores familiarizados con estos lenguajes fueran capaces de tomar el código y en poco tiempo comenzar a programar (45).

Es un lenguaje puramente orientado a objetos, moderno y seguro. Todo el código debe ser incluido en clases, estas derivan de una clase base llamada Objeto, se organizan en espacios de nombres y solo admiten herencia simple. Implementa técnicas de versiones para ayudar a que sus clases evolucionen con el tiempo mientras mantienen la compatibilidad con el código (45).

El código escrito en C# es autocontenido, lo que significa que no necesita de ficheros adicionales. En lo referente a la encapsulación es importante señalar que aparte de los típicos modificadores *public*, *private* y *protected*, C# añade un cuarto modificador llamado *internal*, que puede combinarse con *protected* e indica que al elemento a cuya definición precede sólo puede accederse desde su mismo ensamblado (46).

Otras características que destacan son la orientación a componentes, gestión automática de memoria, seguridad de tipos, instrucciones seguras, sistema de tipos unificado y extensibilidad de tipos básicos y operadores (47).

Java.

Es un lenguaje de programación orientado a objetos, desarrollado a principios de los años 90 por Sun Microsystems. Aunque gran parte de su sintaxis es tomada de C y C++ presenta un modelo orientado a objetos simple y elimina herramientas de bajo nivel que suelen inducir errores como la manipulación directa de punteros (48).

Java adopta el modelo *bytecode* que reduce considerablemente el tamaño de las aplicaciones implementadas, el lenguaje es compilado utilizando la técnica de compilación al instante (del inglés Just In Time, JIT). El compilador JIT realmente lee los *bytecodes* por secciones y los compila de forma interactiva en lenguaje máquina, por lo que el programa puede ejecutarse más rápido (48).

En un sentido estricto, Java no es un lenguaje absolutamente orientado a objetos por motivos de eficiencia. Su código puede ser a veces redundante en comparación con otros lenguajes, debido a las frecuentes declaraciones de tipos, conversiones de tipo manual, no dispone de

operadores de sobrecarga y a una sintaxis relativamente simple.

1.2.3 Entornos de Desarrollo Integrado.

Visual Studio .NET.

Es un Entorno de Desarrollo Integrado (del inglés Integrated Development Environment, IDE) de Microsoft para crear, documentar, ejecutar y depurar programas escritos en una variedad de lenguajes de programación incluidos en la plataforma .NET. Además ofrece herramientas de edición para manipular diferentes tipos de ficheros, es sofisticado y poderoso para la creación de aplicaciones (41).

Algunas de las características más importantes son:

- Editor de Textos: Se encarga de organizar de forma automática las lineas de código, resaltar el inicio y fin de los bloques, aplicar modo de resaltado con diferentes códigos de colores según el grupo al que pertenezca la palabra. También realiza comprobaciones de sintaxis mientras se escribe y subraya el código que puede causar errores en la compilación, implementa completamiento de código automático.
- Editor de vista de diseño: Este editor permite adicionar interfaces de usuario y controles de acceso a datos al proyecto.
- Habilidad de compilar desde dentro del ambiente: En lugar de necesitar administrar un compilador externo desde lineas de comando, el IDE se encarga de pasar todas las lineas de comandos al compilador.
- Depurador integrado: Es muy natural en los programas que el código no se ejecute correctamente la primera o segunda vez que lo intente, para ayudar a encontrar errores en la programación Visual Studio .NET cuenta con un depurador de errores que permite crear puntos de rupturas y así observar el comportamiento de los objetos en tiempo de ejecución (47).

C++ Builder.

Es un entorno de Desarrollo rápido de aplicaciones en lenguaje C++ para Windows, inicialmente propiedad de la empresa Borland, y actualmente de la empresa Embarcadero. Combina la Librería de Componentes Visuales (del inglés Visual Component Library, VCL) y el IDE escrito en Delphi con un moderno compilador de C++. Incluye herramientas que permiten crear formularios, utilizar características de arrastrar-y-soltar componentes de interfaces gráficas y de acceso a datos sobre los formularios y cuenta en su distribución básica con más de 200 componentes (49).

Provee soporte mejorado para ANSI C++, MSBuild, VCL para Web que permite el desarrollo visual de aplicaciones con tecnología AJAX y Soporte de Idioma Integrado. Además soporta las bibliotecas gráficas de Windows Vista y Windows7, permitiendo desarrollar aplicaciones con interfaces multitoques (50).

Qt Creator.

Es un IDE multiplataforma (Windows, Linux/X11 y Mac OS X) que permite compilar código en C++ y brinda herramientas para diseñar y desarrollar aplicaciones basadas en el marco de trabajo (del inglés framework) Qt. El marco de trabajo Qt está diseñado para crear aplicaciones e interfaces de usuarios y desplegarlas en diferentes sistemas operativos basados en entornos de escritorio y en sistemas móviles. Es liberado bajo una licencia Comercial y bajo una Licencia Pública General (del inglés GNU Lesser General Public License, LGPL) (51).

Qt Creator incluye herramientas que permiten elaborar soluciones informáticas siguiendo el ciclo de vida del proyecto, desde la creación de este, hasta su liberación y despliegue en la plataforma selecciona. Una de sus mayores ventajas es que permite a los equipos de desarrolladores compartir proyectos a través de diferentes plataformas en una herramienta común para desarrollar y depurar las soluciones (51).

Sus características más importantes son:

- Sofisticado Editor de Código: Proporciona soporte para editar código en C++ y QML, brinda ayuda de sensibilidad de contexto, completamiento y organización de código, detección de errores de sintaxis y delimitación de bloques, entre otros.
- Control de versiones: Se integra con los sistemas más populares de control de versiones incluidos Git, Subversion, Perforce, CVC y Mercurial.
- Diseñador de Interfaces de Usuario: Proporciona dos editores visuales: un diseñador para construir interfaces de usuario a partir de dispositivos de Qt y un diseñador rápido para desarrollar interfaces de usuario animadas con lenguaje QML.
- Administrador de Proyecto y Compilador: Se encarga de generar todos ficheros necesarios cuando se crea o importa un proyecto, suporta la plataforma de compilación qmake e incluye la plataforma cmake.
- Objetivos de Escritorios y Móviles: Brinda soporte para la construcción y ejecución de aplicaciones basadas en Qt para entornos de escritorio y dispositivos móviles (52).

1.2.4 Selección de los métodos y herramientas.

Teniendo en cuenta las tendencias actuales de desarrollo de software, las características de la investigación, el entorno, la disponibilidad restringida de los especialista que dominan el negocio, la imposibilidad de adquirir herramientas informáticas privativas para la creación del sistema informático y los escasos recursos de hardware, y, partiendo de las características analizadas en los sub-epígrafes Metodologías de desarrollo de software, Lenguajes de programación y Entornos de desarrollo integrados se seleccionan los siguientes métodos y herramientas para desarrollar la solución:

- Metodología AUP: Por brindar un enfoque simple, rápido y organizado del proceso de desarrollo de software, adaptándose a las necesidades del proyecto y permitir, una vez terminado el fujo de trabajo Modelo una independencia marcada entre los desarrolladores y los clientes.

- El lenguaje de programación C++: Por sus potencialidades para el desarrollo de aplicaciones multiplataformas, robustas, con requerimientos de recursos de hardware relativamente bajos, eficiencia en el uso de la RAM y por brindar la posibilidad de utilizar el paradigma de la POO.
- El IDE Qt Creator: Por permitir compilar el lenguaje C++ y utilizar el marco de trabajo Qt que cuenta con una colección amplia de clases y posibilita la creación de interfaces gráficas de altas prestaciones y elevada calidad. Además es multiplataforma y liberado bajo la licencia LGPL.

Conclusiones del Capítulo 1.

Al realizar el estudio del estado del arte no se encontró una herramienta cuyas características y funcionalidades se ajusten a las necesidades requeridas para desarrollar estudios de cuantificación a profundidad sobre el comportamiento visuomotor en pacientes aquejados con SCA2, además ninguna de las herramientas encontradas, por ser privativas permiten su modificación o reutilización de componentes, esto corrobora como posible solución al problema planteado, el desarrollo de un sistema informático que sea capaz de integrar, configurar, aplicar de forma estandarizada y evaluar estos exámenes minimizando el margen de error en los resultados.

Para seleccionar la metodología y herramientas a utilizar en el desarrollo de la solución se realizó un estudio de las más conocidas, enfatizando en las que permiten la construcción de aplicaciones de escritorio, ligeras, rápidas, robustas y con independencia de la plataforma. Además se tuvo en cuenta que contaran con licencias de uso libre y que estuvieran acordes a las necesidades tecnológicas actuales y a las características de la investigación y el entorno. Los métodos y herramientas seleccionadas fueron: metodología de desarrollo AUP, lenguaje de programación C++ y el entorno de desarrollo Qt Creator.

2 Capítulo

Capítulo 2: Descripción de la solución.

La cuantificación de la coordinación visuomotora se rige por los objetivos trazados por el personal médico que la realiza, la selección y configuración de las pruebas que conformarán un estudio, su orden, aplicación y posterior análisis quedarán subordinados a estos.

El proceso de cuantificación (Fig. 5) inicia con la selección o creación del estudio, continúa con el registro de los datos del sujeto a estudiar, la aplicación del estudio a este, el almacenamiento de los resultados y finalmente el análisis de las variables neurológicas de interés.

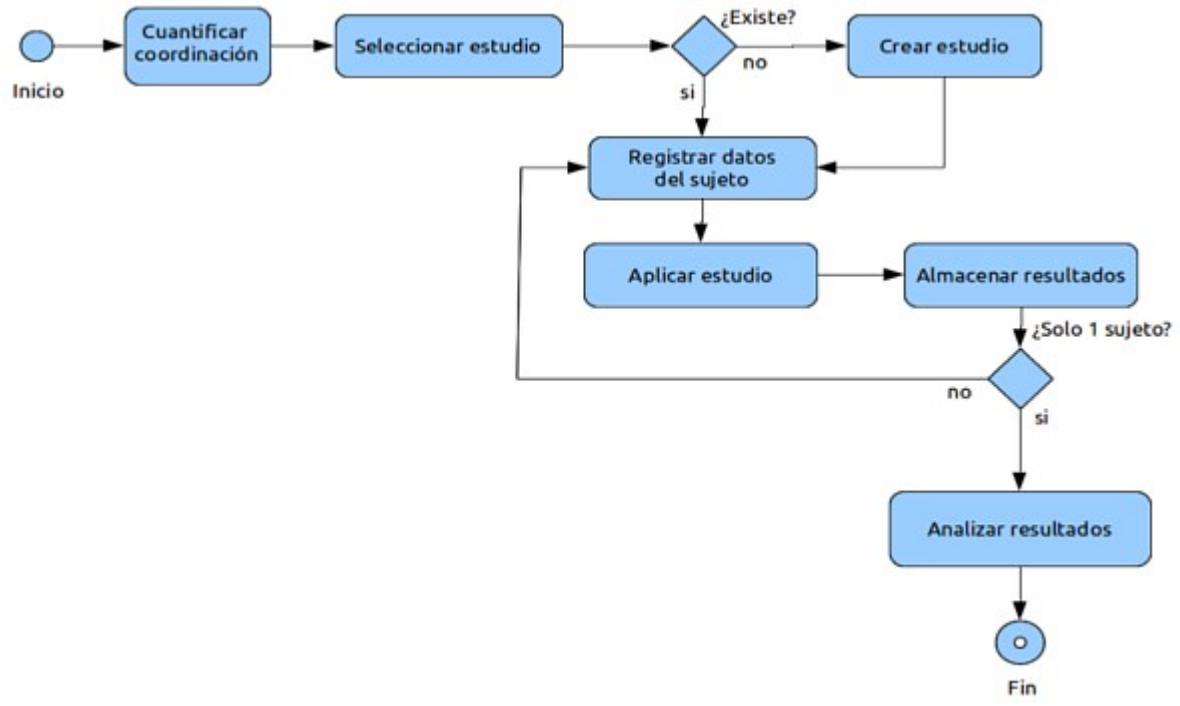

Fig. 5: Diagrama de procesos del negocio.

Un estudio está compuesto por un identificador, un nombre y una o más pruebas

neurológicas que están constituidas por un tipo y una o más fases. Estas últimas almacenan un conjunto de parámetros específicos según el tipo que las define.

Las pruebas seleccionadas para formar parte de la solución informática fueron:

- Prueba Índice punto (Variante de la prueba Índice-nariz estandarizado, adaptada por los especialistas del CIRAH): consiste en intentar alcanzar un punto en la pantalla llevando el dedo índice desde la punta de la nariz hasta este, con la mayor velocidad y precisión posible, para lograr diferentes grados de complejidad puede diseñarse con un punto en posición fija, un punto en posición fija que alterna con posiciones variables o puntos en posiciones variables, al aplicarla se obtienen intervalos de tiempo entre una respuesta y otra, la diferencia en distancia entre el estímulo y la respuesta (dismetría) y las correcciones que ocurren en intervalos de tiempos relativamente cortos. Variables a calcular: dismetría, tiempo y correcciones.
- Prueba Persecución motora[6] (Variante de las pruebas Trazado de círculo y Espiral de Arquímedes , diseñada por los especialistas del CIRAH): consiste en trazar una curva de forma superpuesta a la curva estímulo que se muestra en la pantalla, se puede llevar a grados elevados de complejidad que dependerán de la función matemática que se utilice para generar dicha curva, permite obtener la diferencia del área entre el estímulo y la respuesta, los errores del trazado, velocidad y aceleración del movimiento, las correcciones y el tiempo de ejecución. Variables a calcular: diferencia entre puntos, área, velocidad, aceleración y correcciones.
- Prueba Movimientos alternados (Variante de la prueba Movimientos alternados, adaptada por los especialistas del CIRAH): propone ejercicios de toques alternos a dos botones situados uno al lado del otro en el menor tiempo posible, el grado de dificultad se puede controlar mediante las dimensiones y la separación de los botones,

[6] Con el uso de este examen se han encontrado diferencias estadísticamente significativas entre sujetos sanos y sujetos enfermos para las variables: diferencia entre puntos*, área, velocidad, aceleración y correcciones*. Estos resultados aún sin publicar justifican la inclusión de esta prueba el software a desarrollar.

permite medir los tiempos de respuesta de los movimientos alternados y los errores. Variables a calcular: tiempo y error.

Un sujeto posee características que lo identifican y posibilitan su seguimiento: código, nombre y apellidos, fecha de nacimiento, mano dominante y estado; si el sujeto está en estado presintomático se registra además el diagnóstico clínico y molecular y la familia y generación a la que pertenece; si el sujeto está en estado enfermo se registran todas las características mencionadas anteriormente y la edad de inicio de la enfermedad; además puede ser sometido a varios estudios en diferentes momentos.

De la aplicación de un estudio se almacena su identificador, nombre, mano con que se realizó, fecha en la que se aplica, estado en el que se encuentra el sujeto, edad del sujeto y la relación de las pruebas con sus respectivos resultados, si el sujeto se encuentra en estado enfermo se almacena también el tiempo de evolución de la enfermedad hasta ese momento. Los resultados de los estudios se pueden analizar de forma independiente, en conjunto o comparativa para uno o más sujetos.

2.1 Modelo.

En este flujo de trabajo se realizan un conjunto de actividades que van enfocadas a lograr un entendimiento claro del negocio para determinar una solución viable al problema planteado utilizando representaciones de entidades a nivel conceptual.

2.1.1 Requisitos del software.

A partir del entendimiento del negocio se extrajeron los siguientes requisitos funcionales (RF):

RF-1 Registrar sujeto [código, nombre, apellidos, género (masculino o femenino), manualidad (derecha, izquierda, ambidiestro o mixto), fecha de nacimiento, fecha de registro, edad, estado (sano, presintomático o enfermo), diagnóstico clínico, diagnóstico molecular, edad de inicio de la enfermedad, tiempo de evolución de la

enfermedad, familia, generación y observaciones).

RF-2 Visualizar y generar reporte de sujeto.

RF-3 Modificar sujeto.

RF-4 Crear estudio (identificador, nombre, fecha de creación, listado de pruebas y comentario).

RF-5 Visualizar y generar reporte de estudio.

RF-6 Modificar estudio.

RF-7 Crear prueba (tipo y lista de fases).

RF-8 Visualizar prueba.

RF-9 Modificar prueba.

RF-10 Eliminar prueba.

RF-11 Administrar funciones matemáticas básicas (seno, coseno, tangente, raíz cuadrada, etc.).

RF-12 Ejecutar estudio.

RF-13 Almacenar resultado de estudio [identificador, nombre, mano usada (derecha o izquierda), fecha de aplicación, estado del sujeto (sano, presintomático o enfermo), edad del sujeto, tiempo de evolución de la enfermedad y los resultados de las pruebas].

RF-14 Explorar directorios en busca de sujetos.

RF-15 Filtrar sujetos [género (masculino o femenino), manualidad (derecha, izquierda, ambidiestro o mixto), estado (sano, presintomático o enfermo), tiempo de evolución de la enfermedad, edad del sujeto, diagnóstico clínico y molecular, familia, generación y estudio aplicado].

RF-16 Generar reporte del resultado de un estudio perteneciente a un sujeto.

RF-17 Generar reporte del resultado de varios estudios pertenecientes a un sujeto.

RF-18 Generar reporte fusionado del resultado de varios estudios con un identificador común, pertenecientes a un sujeto.

RF-19 Generar reporte fusionado del resultado de varios estudios con un identificador común, pertenecientes a varios sujetos.

RF-20 Generar reporte comparativo del resultado de un estudio entre dos grupos de sujetos.

RF-21 Generar reporte comparativo del resultado de un estudio entre un grupo de sujetos y un sujeto.

RF-22 Generar reporte del resultado de un estudio para cada uno de los sujetos a los que se le haya aplicado el estudio correspondiente.

RF-23 Realizar cálculos estadísticos de variables de los resultados de estudios (media, mínimo, máximo y desviación estándar).

RF-24 Exportar reportes (formato PDF, formato CSV y formato HTML).

RF-25 Imprimir reportes.

RF-26 Configurar parámetros para la ejecución de los estudios [color de fondo, color del estimulo, color de la respuesta, mostrar puntero (si o no), estilo de linea, retardo y sensibilidad].

RF-27 Configurar parámetros del sistema (resolución de pantalla).

Además de los requisitos funcionales se extrajo el siguiente listado de requisitos no funcionales (RNF):

RNF-1 El diseño debe ser sencillo e intuitivo.

RNF-2 La interfaz debe ser amigable con colores tenues, agradables a la vista.

RNF-3 El sistema debe permitir la ejecución de funcionalidades mediante combinaciones de teclas.

RNF-4 Las funcionalidades principales del sistema deben ser visibles en todo momento.

RNF-5 El sistema debe informar sobre el estado de la información mediante alertas para evitar pérdidas de esta.

RNF-6 El sistema debe contar con un manual de ayuda.

RNF-7 El sistema debe ser preciso en la captura y análisis de los datos de los estudios.

RNF-8 El sistema debe ejecutar sus funcionalidades con tiempos de respuestas aceptables, dependiendo del hardware disponible.

RNF-9 El sistema debe ser compatible para compilación en diferentes plataformas (Windows, Linuxs, Mac OS.).

RNF-10 El sistema debe funcionar en una computadora con un procesador Pentium III, 128 MB de RAM, 60 GB de disco duro, 64 MB de memoria de vídeo o características superiores, a una resolución de pantalla de 800X600 píxeles o superior y un monitor táctil adicional.

2.1.2 Casos de uso del sistema.

El diseño de los casos de uso del sistema (Fig. 6) partió de los requisitos funcionales extraídos del negocio. Los actores identificados fueron el Técnico que es el responsable de operar todas las funcionalidades del sistema y el Sujeto que interactúa con el sistema durante el proceso de ejecución de estudios.

Casos de uso del sistema:

CUS-1 Gestionar sujeto: El técnico registra, visualiza o actualiza un sujeto.

CUS-2 Gestionar estudio: El técnico crea, visualiza o actualiza un estudio.

CUS-3 Gestionar prueba: El técnico crea, visualiza, actualiza o elimina una prueba.

CUS-4 Administrar funciones matemáticas: El técnico crea, modifica o visualiza una función matemática con el objetivo de gestionar una prueba de tipo Persecución motora .

CUS-5 Ejecutar estudio: El técnico aplica un estudio a un sujeto, el sujeto completa todas las pruebas impuestas y al concluir, si se desarrolló de forma correcta, el técnico guarda los resultados del estudio, si no, el técnico repite el estudio al sujeto .

CUS-6 Buscar sujetos: El técnico realiza búsquedas de sujetos por diferentes directorios, de forma simple o compuesta .

CUS-7 Filtrar sujetos: El técnico filtra sujetos por diferentes criterios .

CUS-8 Generar reportes: El técnico genera reportes de los resultados de los estudios aplicados a uno o varios sujetos. Los reportes se pueden generar por estudios o por sujetos y pueden ser simples, generales o comparativos. Los reportes se pueden imprimir o ser exportados a diferentes formatos .

CUS-9 Configurar sistema: El técnico configura diferentes parámetros del sistema y de la ejecución de las pruebas .

Nota: Para ver la descripción en formato de alto nivel de los casos de uso del sistema consultar el Anexo 1.

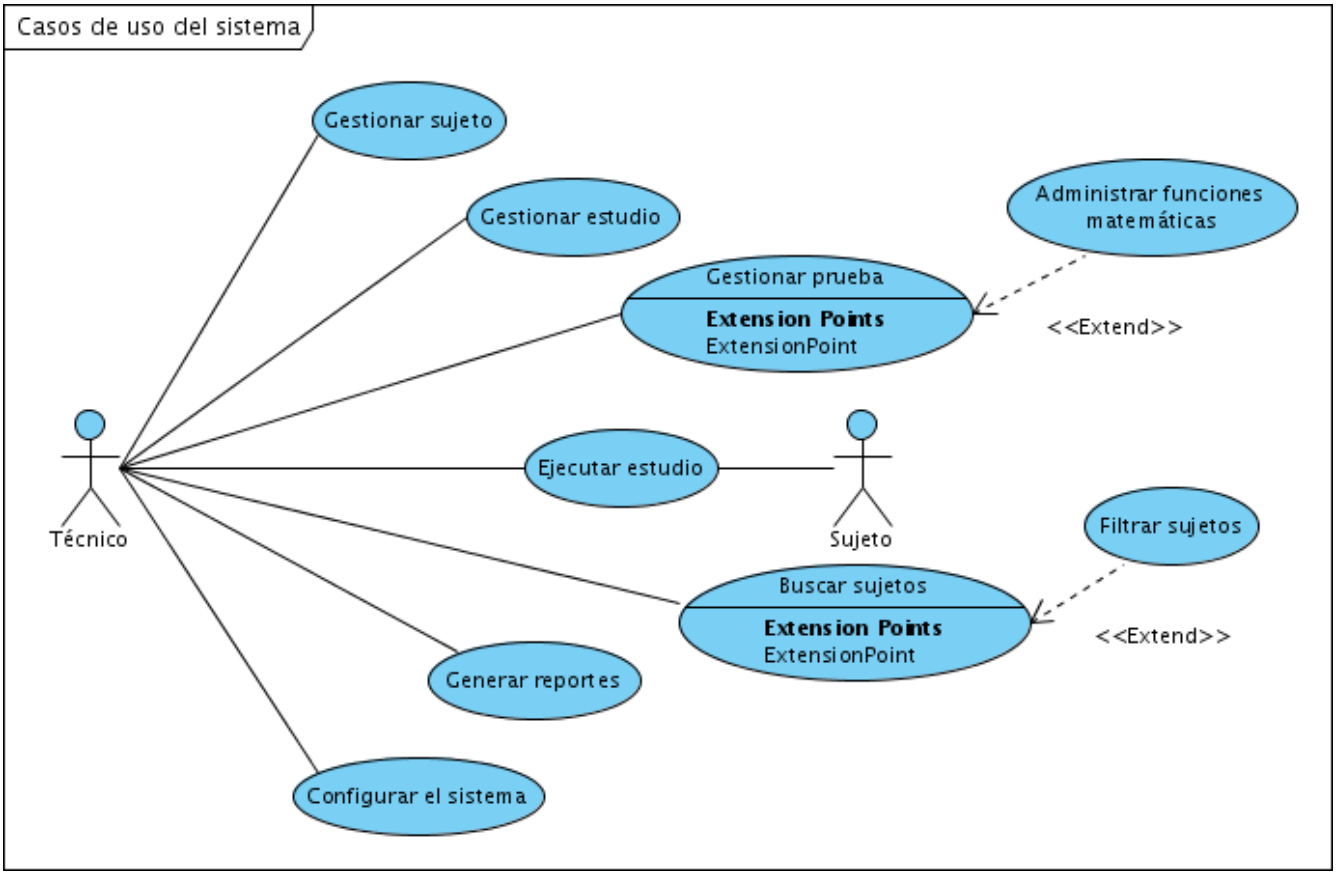

Fig. 6: Diagrama de Casos de uso del sistema.

2.1.3 Arquitectura.

El estilo arquitectónico seleccionado fue el estilo de Llamada y Respuesta y dentro de este, la variante Modelo-Vista-Controlador (MVC). Con esta variante se logra obtener un marco de referencia para el desarrollo organizado del software y la independencia entre el modelo, el controlador y las vistas, de tal forma que es posible desarrollar las capas por separado y realizar una gestión óptima de los cambios. Permite además la flexibilidad y escalabilidad de la solución.

La organización general de las capas quedó de la siguiente forma (Fig. 7):

- Vista (VIEW): Se encarga de gestionar todas la interfaces visuales del sistema.

- Controlador (CORE, REPORT y GRAPH): Gestiona el manejo de las entidades del sistema, realiza los cálculos matemáticos, controla la ejecución de los estudios y la captura de los resultados de la aplicación de estos, y gestiona la generación de reportes.
- Modelo (DAO): Se encarga de leer y escribir los datos de las entidades del sistema.
- Framework Qt: Permite crear interfaces gráficas de usuarios con altas prestaciones y elevada calidad y brinda un amplio conjunto de clases escritas en C++ que permiten un desarrollo rápido y eficiente.
- GSL: Colección robusta de rutinas para análisis numérico GNU Librería Científica (del inglés GNU Scientific Library), escrita en lenguaje C (53).

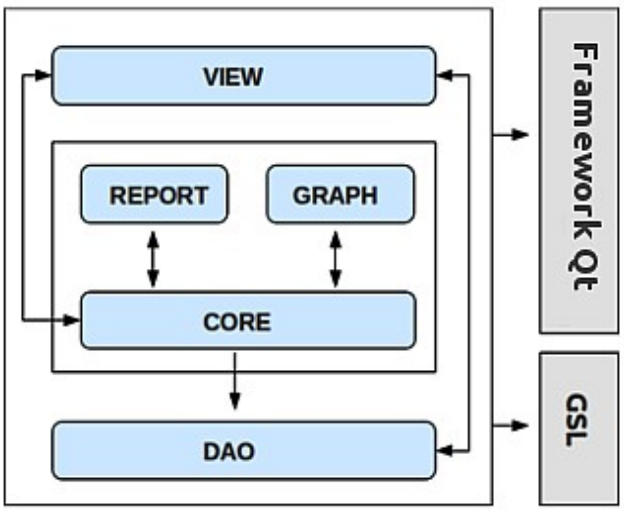

Fig. 7: Modelo arquitectónico del sistema.

Paquetes del sistema

Las clases del sistema se organizaron en paquetes con el objetivo de transmitir detalladamente la composición de las capas declaradas por el modelo arquitectónico (Fig. 8).

- Paquete VIEW: Este paquete está integrado por las clases que gestionan las interfaces gráficas de usuario, accede a las funcionalidades de los paquetes CORE, REPORT, GRAPH y DAO.
- Paquete CORE: Este paquete está integrado por las clases que gestionan y definen las estructuras de los sujetos, estudios, pruebas, fases y resultados de estudios, además se encarga de los cálculos matemáticos, conversiones y manejo de funciones matemáticas. Este paquete accede a las funcionalidades del paquete DAO.
- Paquete GRAPH: Este paquete está integrado por las clases que ejecutan los estudios y capturan los resultados de estos, accede a las funcionalidades del paquete CORE.
- Paquete REPORT: Este paquete está integrado por las clases que manejan las funcionalidades de generar reportes, exportar a diferentes formatos e imprimir las salidas de estos, accede a las funcionalidades del paquete CORE.
- Paquete DAO: Este paquete está integrado por las clases que manejan la lectura y escritura en ficheros de las entidades sujeto, estudio y resultados de estudios.

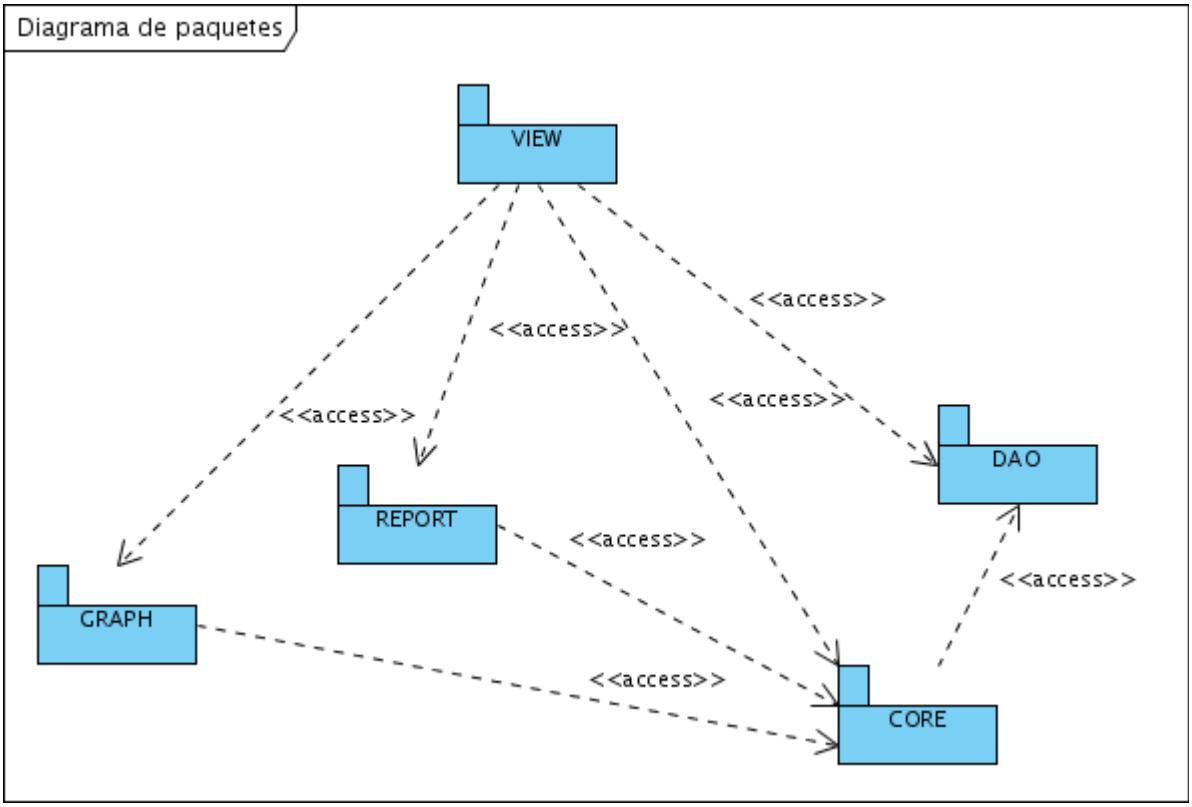

Fig. 8: Diagrama de paquetes de clases del sistema.

Nota: Para ver los diagramas de las clases del sistema por paquetes consultar el Anexo 2.

Subsistemas.

Con el objetivo de simplificar el problema principal, se dividió en problemas más pequeños con metas específicas, alcanzables en períodos de tiempos relativamente cortos. Estos problemas más pequeños se modelaron en forma de subsistemas: Subsistema Sujetos, Subsistema Diseño, Subsistema Ejecución, Subsistema Análisis y Subsistema Operaciones Matemáticas (Fig. 9).

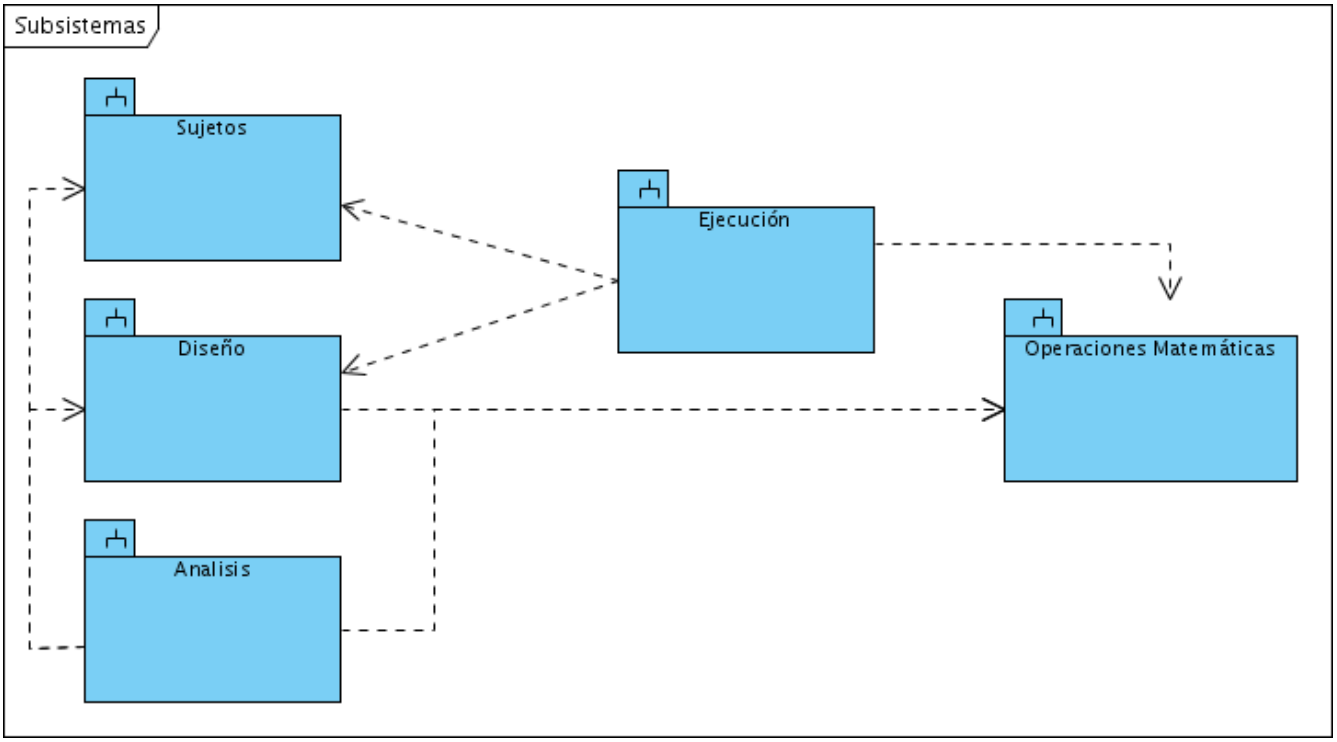

Fig. 9: Subsistemas de la aplicación.

2.2 Implementación.

La implementación del sistema se realizó de forma organizada, siguiendo el orden de prioridad de los casos de uso del sistema y guiada por pruebas unitarias. A continuación se detallan los elementos más significativos desarrollados durante el proceso de implementación.

2.2.1 Intérprete de funciones matemáticas.

Este intérprete es el encargado de crear, modificar y evaluar las funciones matemáticas que son utilizadas en las pruebas de Persecución motora, se basa en una parte de la teoría de compiladores al utilizar un Analizador lexicográfico y un Analizador sintáctico. Para ver los detalles del diseño de las clases que lo componen consultar el Anexo 3.

El alfabeto definido para el intérprete se representa como:

$$\sum \{a, b, c, \ldots, z, A, B, C, \ldots, Z, 0, 1, 2, \ldots, 9, +, -, *, /, \wedge, (,)\}$$

donde ∑ es el alfabeto y los valores dentro de las llaves son los símbolos que lo conforman. El lenguaje para representar una función matemática está dado por el siguiente conjunto de reglas:

- Las variables se definen como la combinación de uno o más caracteres que pertenecen al conjunto “a – z, A – Z”.
- Los número se definen como la combinación de uno o más caracteres que pertenecen al conjunto “0 – 9”.
- Los operadores matemáticos se definen como la ocurrencia única de uno de los caracteres que pertenecen al conjunto “+, -, *, /, ^”.
- Las palabras reservadas del lenguaje son: *sen*, *cos*, *log*, *sqrt*, *senh*, *cosh*, *asen*, *acos*, *acosh* y *asenh*. Representan funciones matemáticas.
- Una función matemática puede estar formada por variables, números, operadores, paréntesis, palabras reservadas y funciones matemáticas.
- En la expresión tiene que existir una única variable, sin importar cuantas veces se repita.
- La expresión no puede comenzar o finalizar con ninguno de los operadores matemáticos.
- En la expresión por cada paréntesis que se abra tiene que haber uno que lo cierre, y no pueden estar vacíos “()”.
- Las funciones matemáticas representadas por las palabras reservadas del lenguaje tienen que estar escritas en el siguiente formato: *palabra_reservada*(función).

El Analizador lexicográfico realiza una lectura lineal de izquierda a derecha del fujo de

caracteres que componen el código fuente escrito para representar una función matemática, comprueba que estos pertenezcan al alfabeto y los agrupa en “tokens” o símbolos que son secuencias de caracteres con un significado. Los “tokens” definidos son: número, operador, variable, palabra reservada y paréntesis. El Analizador sintáctico comprueba a partir del listado de “tokens” que las reglas definidas para el lenguaje se cumplan.

Para evaluar la función en un punto se agrupan en un listado todos los operadores matemáticos y las palabras reservadas según el orden en que aparecen en el listado de los “tokens” y en un listado de valores los números y las variables sustituidas por el valor a evaluar. De esta forma se garantiza que el valor para cada operador unario se encuentre en la misma posición que dicho operador y que los valores para cada operador binario se encuentren, el primero en la misma posición y el segundo en la posición siguiente. El cálculo se realiza respetando las reglas matemáticas que definen el orden de las operadores.

Cada expresión que aparece dentro de un paréntesis se maneja como una función independiente y se calcula su valor con una llamada recursiva almacenándose en el listado de valores.

La interfaz gráfica que permite la interacción con el intérprete para crear y modificar funciones matemáticas (Fig. 10) está equipada con técnicas de resaltado para diferenciar las palabras reservadas, los números, operadores y variables. Además informa si la expresión está escrita con la sintaxis correcta.

Fig. 10: Interfaz gráfica para la creación y edición de funciones matemáticas.

2.2.2 Procesador de señales de movimiento.

El Procesador de señales de movimiento es el encargado de calcular las diferencias entre una señal de movimiento de estímulo y una señal de movimiento de respuesta a partir de las

distancias entre sus puntos y los valores de sus áreas. Además permite calcular la velocidad, aceleración y correcciones para la señal de movimiento de respuesta. Los datos que permiten realizar este procesamiento son capturados durante la ejecución de las pruebas de Persecución motora. Para ver los detalles del diseño de las clases que lo componen consultar el Anexo 4.

De ambas señales se tiene un conjunto de puntos en el eje de coordenadas “x;y”, y de la señal de respuesta se tiene, para cada punto en el espacio, el tiempo de demora en el desplazamiento desde el punto anterior hasta él.

El primer parámetro de comparación es la diferencia punto a punto o error entre las dos señales, por los datos se tiene el listado de los puntos ordenados en el eje de las “x” de la señal de estímulo y el listado de los puntos ordenados según la secuencia de construcción de la señal de respuesta. A partir de estos listados se calcula la distancia entre el primer punto de ambas señales y así sucesivamente para los *N* puntos hasta que una de las dos listas llegue a su fin. La media de los resultados de la comparación punto a punto junto con la desviación estándar permiten caracterizar la diferencia entre ambas señales, así, a medida que estos valores se aproximen a cero, la diferencia y variación de la repuesta respecto al estímulo será menor.

Para realizar los cálculos de área de la señal de estímulo y los cálculos de área, velocidad y aceleración de la señal de respuesta es necesario encontrar la función matemática que las caracteriza. El método de interpolación de polinomios permite obtener un polinomio aproximado que pasa por todos los puntos de la muestra sin perder los datos originales, específicamente se utiliza la interpolación por spline cúbica que suaviza el polinomio haciéndolo continuo y diferenciable de manera que su primera y segunda derivadas sean continuas (54) , este método se encuentra disponible en la colección de rutinas matemáticas GSL.

El cálculo de las áreas de las dos señales se realiza mediante la integral definida desde el punto inicial hasta el punto final de la señal, este método permite encontrar el área de cualquier región definida por una o varias curvas sin importar la forma que tenga (55), esta

funcionalidad se encuentra disponible en la colección de rutinas matemáticas GSL.

$$\textit{área debajo de la curva}_{\textit{desde a hasta b}} = \int_a^b f(x)\,dx$$

Encontrar la función para la señal de respuesta no es un proceso trivial, hay que tener en cuenta que el requisito para realizar la interpolación de un conjunto de puntos es que estos sean pares ordenados en el eje de las “x” y muchas veces esto no se cumple al ser ordenados según el proceso de construcción de la señal.

Para solucionar este problema se aplica un algoritmo basado en el valor medio, por conveniencia se denota:

$$\textit{punto en el espacio} \rightarrow (X;Y)$$

Este algoritmo consiste en buscar los segmentos de retroceso en la señal de desplazamiento –de respuesta– y para cada segmento encontrar la media de las Y del conjunto de sus puntos y sustituir estos por el punto formado por la media de las X de los puntos que delimitan el segmento y por la media de las Y pertenecientes al segmento en cuestión. Además hay que actualizar los tiempos de desplazamiento, para esto se eliminan los valores que corresponden al segmento suprimido y para el punto nuevo se le asigna el valor medio de los tiempos que corresponden a los puntos que delimitan el segmento.

Cuando la longitud de un segmento de retroceso sobrepasa cierto umbral configurable es tomado como una corrección.

La velocidad y aceleración se pueden calcular como la primera y segunda derivadas de la ecuación del desplazamiento en función del tiempo (56). Para convertir la señal de respuesta –después de aplicarle el algoritmo basado en el valor medio– a una señal de desplazamiento en función del tiempo se calcula la distancia recorrida y el tiempo de demora entre un punto y otro para toda la señal, con estos datos es posible crear el par ordenado “tiempo;desplazamiento”. Los pares de puntos son interpolados y posteriormente se les aplica la primera y segunda derivada en cada punto desde el inicio hasta el final, del conjunto de valores obtenidos se extrae entonces la velocidad y aceleración media.

$$\vec{v} = \frac{ds}{dt}$$

$$\vec{a} = \frac{dv}{dt}$$

Las funcionalidades de este procesador son consumidas por el Asistente para generar reportes cuando se enfrenta a la gestión de los datos de las pruebas de Persecución motora.

2.2.3 Motor gráfico.

El Motor gráfico gestiona las operaciones de dibujo involucradas en la aplicación de los estudios, permite la configuración de los colores de fondo, de estímulo y de respuesta y del grosor de las líneas. Para ver los detalles del diseño de las clases que lo componen consultar en el Anexo 2 el Diagrama de clases, paquete GRAPH.

Para generar los gráficos, el motor hace uso de la clase *QPainter* que proporciona funciones altamente optimizadas con las que se pueden dibujar desde líneas hasta figuras complejas, también consume funcionalidades de las clases *QPaintEvent*, *QKeyEvent* y *QMouseEvent*, todas estas clases pertenecen al Framework Qt.

Cada una las pruebas requiere de métodos diferentes para mostrar los estímulos y la retroalimentación de las respuestas (Fig. 11), pero es común a todas las conversiones de los datos de milímetros a píxeles y viceversa, necesarias para graficar con exactitud los estímulos y para almacenar los resultados de las respuestas.

$$píxeles = milímetros / (a / rh)$$

$$milímetros = píxeles * (a / rh)$$

Donde *a* es el ancho de la pantalla en milímetros y *rh* es la resolución de pantalla en el eje horizontal.

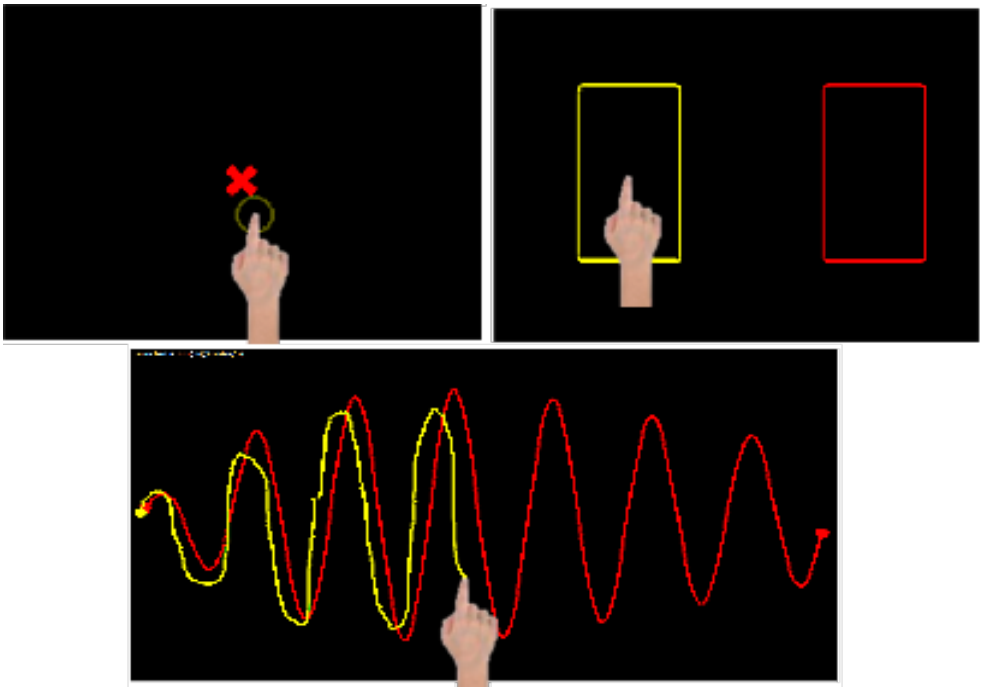

Fig. 11: Estímulo y retroalimentación en la ejecución de las pruebas.

Índice punto.

Para la prueba de Índice punto las posiciones de aparición de los puntos de estímulos se encuentran almacenados en un listado, el motor realiza el dibujo de cada uno de ellos de forma similar a una “X”, para lograrlo dibuja dos rectas dadas por los puntos:

$P_{1,1}(x-g;y-g),\ P_{1,2}(x+g;y+g)$

$P_{2,1}(x-g;y+g),\ P_{2,2}(x+g;y-g)$

donde *g* es el grosor de la línea de dibujo.

La retroalimentación de la respuesta se grafica mediante un conjunto de círculos concéntricos que dan la impresión de animación en la posición de respuesta del sujeto.

Persecución motora.

La prueba de Persecución motora tiene tres variables fundamentales: la *función* que define la forma del estímulo y los valores de *inicio* y *fin* que definen donde comienza y termina el

estímulo. Para obtener los puntos de estímulos a graficar se evalúa la *función* para los *i* valores del intervalo cerrado [*inicio, fin*]. Las gráficas para esta prueba se dibujan en el primer cuadrante del eje de coordenadas “x;y”, pero las pantallas representan el cuarto cuadrante, para resolver este problema hay que desplazar el eje de coordenadas hacia la posición:

$$\left(0;-\frac{resolución-vertical}{2}\right),$$

esto significa que los valores de “y” resultantes de la evaluación de la *función* hay que actualizarlos siguiendo la fórmula:

$$y_{corregida}=-y_{actual}+\frac{resolución-vertical}{2},$$

luego de estos ajustes solo queda enlazar con una recta cada uno de los puntos para obtener la gráfica de la *función*.

La retroalimentación de las respuesta se realiza graficando la unión con una recta de cada uno de los puntos generados por la respuesta del sujeto.

Movimientos alternados.

La prueba de Movimientos alternados tiene tres variables importantes: el *alto* y *ancho* de los dos botones y la *distancia* entre ellos. Los dos botones se encuentran uno al lado del otro en el eje horizontal a una distancia *x* de separación, alineados respecto al centro de la pantalla, para lograr graficarlos es necesario encontrar el punto donde se dibujará la esquina superior izquierda de ambos:

$$x_{botón\ izquierdo}=\frac{resolución-horizontal}{2}-\left(\frac{distancia}{2}+ancho\right),$$

$$x_{botón\ derecho}=\frac{resolución-horizontal}{2}+\frac{distancia}{2},$$

$$y_{ambos}=\frac{resolución-vertical}{2}-\frac{ancho}{2},$$

una vez que se tienen estos datos se dibuja los rectángulos con las dimensiones específicas.

La retroalimentación de las respuestas se realiza añadiéndole al botón tocado un relieve que simula un estado de presionado.

2.2.4 Asistente para generar reportes.

El Asistente para generar reportes es una herramienta que guía paso a paso el proceso de construcción de reportes. Los reportes se pueden generar basados en estudios o sujetos y en dependencia de los datos con que se trabaje se pueden visualizar en modo simple, fusionado o comparativo. Para ver los detalles del diseño de las clases que lo componen consultar en el Anexo 2 el Diagrama de clases, paquete VIEW (parte 3).

El asistente permite seleccionar con precisión los estudios, pruebas, variables estadísticas, campos de información de los sujetos y variables de las pruebas que se incluirán en el reporte, así como el estilo de visualización de los resultados.

Para los reportes por estudios, en el caso en que estén involucrados varios sujetos, el asistente muestra solo los estudios que son comunes para todos y de existir sujetos que no tengan ningún estudio que coincida con los demás son retirados del análisis.

La vista de reportes comparativos tiene una alta importancia al permitir comparar el comportamiento entre una muestra y un sujeto y entre dos muestras. Los reportes por sujetos permiten obtener los datos en un formato lineal, óptimo para someterlo a otras pruebas estadísticas presentes en herramientas especialidades con estos fines.

Es posible generar nuevos reportes sin necesidad de cerrar los resultados antiguos que se pueden enviar a un segundo plano y después recuperarlos, esto permite realizar comparaciones avanzadas entre varios reportes. Los resultados de los reportes pueden ser impresos directamente si se tiene una impresora configurada en el sistema operativo, además pueden ser exportados a formato PDF, CSV y HTML.

Conclusiones del Capítulo 2.

Al realizar los procesos del flujo de trabajo modelo se logró un completo entendimiento de los procesos del negocio, posibilitando la extracción satisfactoria de los requisitos funcionales, a partir de los cuales se construyó el modelo de casos de uso del sistema y posteriormente el diseño de las clases del sistema organizadas en paquetes.

La arquitectura diseñada permitió que la implementación del sistema se realizara por partes, de forma organizada y rápida, logrando obtener un sistema con alta cohesión y bajo acoplamiento entre los diferentes subsistemas. Los requisitos no funcionales identificados permitieron que el software desarrollado cuente con un conjunto de características de usabilidad que le reportan un alto valor agregado.

3

Capítulo

Capítulo 3: Validación de la solución.

Existen diferentes técnicas para corroborar la calidad de una solución informática durante su proceso de construcción e implantación, por lo general el objetivo principal que persiguen es demostrar que el sistema desarrollado cumple con los requisitos funcionales y no funcionales extraídos durante el flujo de trabajo modelo y que la arquitectura propuesta fue la correcta evaluando la estabilidad, usabilidad, fiabilidad y precisión del software.

En este capítulo es de interés además demostrar que los datos recibidos por la pantalla táctil son correctos y que el sistema es sostenible en las dimensiones administrativa, socio-humanista, ambiental y tecnológica.

3.1 Pruebas unitarias.

La metodología AUP propone un desarrollo dirigido por pruebas, que permiten evacuar de forma rápida y con el menor número de modificaciones posibles cualquier error detectado. Estas pruebas son automatizadas, unitarias y se escriben antes de desarrollar el código que dará solución al problema del cual surgió dicha prueba, este problema está determinado por uno o varios requisitos funcionales, de esta forma se garantiza que se desarrollen todos los requisitos funcionales con la calidad necesaria.

Para desarrollar las pruebas unitarias se utilizó la librería *QTestLib* que se encuentra en el marco de trabajo Qt.

3.1.1 Ejecución de exámenes y captura de los datos.

Durante los procesos de ejecución de exámenes y captura de los datos se ejecutan un conjunto de algoritmos críticos, de los cuales es necesario evaluar su fiabilidad y precisión para controlar la calidad de los datos que se almacenarán.

En la preparación de los datos para la ejecución de las pruebas intervienen el algoritmo de conversión de milímetros a píxeles y los algoritmos pertenecientes al Intérprete de funciones matemáticas, mientras que en la captura de los datos interviene el algoritmo de conversión de píxeles a milímetros.

Prueba unitaria a los algoritmos de conversión de milímetros a píxeles y píxeles a milímetros.

A partir de las fórmulas para realizar la conversión de milímetros a píxeles y viceversa analizadas en el sub-epígrafe Motor gráfico se diseñó una prueba unitaria con cuatro escenarios distintos para cada conversión, donde el valor actual[7] fue calculado de forma manual y el esperado[8] fue la llamada a los algoritmos encargados de estas funcionalidades.

Milímetros	a	rh	Valor actual	Valor esperado
3	376,174	1280	10,208	10,208
8	376,174	1280	27,2214	27,2214
3	304,80	1024	10,0787	10,0787
8	304,80	1024	26,8766	26,8766

Tabla 1: Datos de la prueba unitaria al algoritmo de conversión de milímetros a píxeles.

Píxeles	a	rh	Valor actual	Valor esperado
15	376,174	1280	4,4083	4,4083
22	376,174	1280	6,4655	6,4655
15	304,80	1024	4,4648	4,4648
22	304,80	1024	6,5484	6,5484

Tabla 2: Datos de la prueba unitaria al algoritmo de conversión de píxeles a milímetros.

En ambas tablas se puede apreciar que el valor esperado se corresponde con el valor actual

[7] Es el valor que se conoce como correcto.

[8] Es el valor que arroja el algoritmo probado.

para los juegos de datos mostrados, haciéndose extensible este resultado a todas las posibles combinaciones. Por tanto queda demostrado que los algoritmos de conversión de milímetros a píxeles y de píxeles a milímetros son precisos y fiables.

Prueba unitaria a los algoritmos pertenecientes al Intérprete de funciones matemáticas.

Los algoritmos pertenecientes al Intérprete de funciones matemáticas tienen una importancia crítica en la construcción de los estímulos y en el análisis de los datos de las pruebas de Persecución motora, para controlar la fiabilidad y precisión de estos se diseñó una prueba unitaria con tres escenarios distintos para juegos de datos de veinte números generados aleatoriamente.

Para generar los números aleatorios y encontrar los valores actuales en cada uno de los escenarios se utilizó la herramienta matemática Euler (57), se debe tener en cuenta que el Intérprete de funciones matemáticas durante la evaluación de las funciones trigonométricas multiplica los argumentos de estas por Pi/180 (conversión de grados a radianes).

Para encontrar los valores esperados en cada uno de los escenarios se realizó la llamada al conjunto de algoritmos que posibilitan la evaluación de las funciones matemáticas.

Funciones	Números	Valores actuales	Valores esperados
100*sen(x)	[46, 2, 60, 10, 61, 21, 96, 84, 32, 56, 91, 99, 47, 18, 42, 72, 24, 83, 44, 37]	[71.934, 3.48995, 86.6025, 17.3648, 87.462, 35.8368, 99.4522, 99.4522, 52.9919, 82.9038, 99.9848, 98.7688, 73.1354, 30.9017, 66.9131, 95.1057, 40.6737, 99.2546, 69.4658, 60.1815]	[71.934, 3.48995, 86.6025, 17.3648, 87.462, 35.8368, 99.4522, 99.4522, 52.9919, 82.9038, 99.9848, 98.7688, 73.1354, 30.9017, 66.9131, 95.1057, 40.6737, 99.2546, 69.4658, 60.1815]
20000/x*sen(2x)	[77, 37, 22, 1, 16, 35, 40, 35, 34, 16, 58, 83, 90, 91, 9, 38, 16, 23, 94, 44]	[113.863, 519.601, 631.508, 697.99, 662.399, 536.967, 492.404, 536.967, 545.402, 662.399, 309.929, 58.2944, 2.72135e-14, -7.67022, 686.704, 510.682, 662.399, 625.513, -29.6113, 454.269]	[113.863, 519.601, 631.508, 697.99, 662.399, 536.967, 492.404, 536.967, 545.402, 662.399, 309.929, 58.2944, 2.72135e-14, -7.67022, 686.704, 510.682, 662.399, 625.513, -29.6113, 454.269]
x*sen(2x)/cosh(x/8)	[13, 5, 18, 77, 23, 97, 3, 19, 64, 23, 9, 89, 98, 21, 55, 49, 57, 81, 98, 26]	[5.69653, 0.868189, 10.572, 33.2838, 16.524, -22.9506, 0.313579, 11.6875, 49.945, 16.524, 2.78062, 3.04841, -26.4066, 14.037, 51.3132, 48.2472, 51.672, 24.6446, -26.4066, 20.4554]	[5.69653, 0.868189, 10.572, 33.2838, 16.524, -22.9506, 0.313579, 11.6875, 49.945, 16.524, 2.78062, 3.04841, -26.4066, 14.037, 51.3132, 48.2472, 51.672, 24.6446, -26.4066, 20.4554]

Tabla 3: Datos de la prueba unitaria a los algoritmos pertenecientes al Intérprete de funciones matemáticas.

En esta tabla se puede apreciar que los valores esperados se corresponden con los valores actuales para cada uno de los escenarios de la prueba, haciéndose extensible estos resultados a todas las funciones aceptadas por la sintaxis definida en el Intérprete de funciones matemáticas evaluadas en cualquier valor del eje de las “x”. Por tanto queda demostrado que los algoritmos pertenecientes al Intérprete de funciones matemáticas son precisos y fiables.

3.1.2 Procesamiento de datos.

En el procesamiento de los datos intervienen un conjunto de algoritmos críticos para el

resultado de los procesos de análisis, de los cuales es necesario evaluar su precisión y fiabilidad.

Durante el análisis de las pruebas, se realizan cálculos de distancia, área, velocidad y aceleración, así como interpolación y evaluación de polinomios.

Prueba unitaria al algoritmo de cálculo de distancia.

El cálculo de distancia de un punto a otro se utiliza en el análisis de los datos de las pruebas Índice punto y Persecución motora, para controlar la fiabilidad y precisión de los resultados arrojados por este algoritmo, se diseñó una prueba unitaria con cuatro escenarios distintos. Los valores actuales se calcularon utilizando la herramienta Euler y los valores esperados se obtuvieron mediante la llamada al algoritmo de cálculo de distancia.

Punto 1	Punto 2	Valor actual	Valor esperado
(2;7)	(12;2)	11.1803	11.1803
(10;19)	(4;7)	13.4164	13.4164
(22;36)	(9;45)	15.8114	15.8114
(0;1)	(11;0)	11.0445	11.0445

Tabla 4: Datos de la prueba unitaria al algoritmo de cálculo de distancia.

En la tabla se puede apreciar que el valor esperado se corresponde con el valor actual para los pares de puntos mostrados, haciéndose extensible este resultado a todos los posibles pares de puntos. Por tanto queda demostrado que el algoritmo de cálculo de distancia es preciso y fiable.

Pruebas unitarias a los algoritmos pertenecientes al Procesador de señales de movimiento.

Los algoritmos pertenecientes al Procesador de señales de movimiento tienen una

importancia crítica en el análisis de los datos de las pruebas de Persecución motora, para controlar la fiabilidad y precisión de estos, se diseñaron dos pruebas unitarias.

Prueba 1: Para evaluar el funcionamiento de los algoritmos de interpolación y evaluación se diseñó una prueba con dos juegos de datos distintos, cada uno formado por diez puntos generados de forma aleatoria, el polinomio resultante es evaluado en quince puntos que se encuentran dentro del intervalo definido por el mínimo y máximo valor en el eje “x” de los puntos generados. Los valores actuales se calcularon mediante interpolación con el método spline cúbica utilizando la herramienta Euler –como condición necesaria para realizar la interpolación los puntos deben estar ordenados de de forma ascendente en el eje “x” (53)– y los valores esperados se obtuvieron mediante el conjunto de algoritmos que conforman el Procesador de señales de movimiento.

Puntos def [9].	**Puntos eval [10].**	**Valor actual**	**Valor esperado**
[(1;3), (4;10), (5;4), (6;2), (10;-5), (11;-6), (12;-8), (14;-3), (17;1), (19;5)]	[1, 2, 3, 5, 6, 8, 9, 11, 12, 14, 15, 16, 17, 18, 19]	[3, 8.56137, 11.7017, 4, 2, -1.58184, -3.61044, -6, -8, -3, -0.929067, 0.0466666, 1, 2.7374, 5]	[3, 8.56137, 11.7017, 4, 2, -1.58184, -3.61044, -6, -8, -3, -0.929067, 0.0466666, 1, 2.7374, 5]
[(10;100), (12;77), (14;69), (15;23), (17;10), (20;2), (21;-3), (27;-5), (31;1), (35;10)]	[11, 13, 14, 19, 20, 22, 26, 27, 28, 29, 30, 32, 33, 34, 35]	[83.513, 82.336, 69, 7.76297, 2, -6.23059, -6.42036, -5, -3.60461, -2.20698, -0.705866, 2.98225, 5.19399, 7.55875, 10]	[83.513, 82.336, 69, 7.76297, 2, -6.23059, -6.42036, -5, -3.60461, -2.20698, -0.705866, 2.98225, 5.19399, 7.55875, 10]

Tabla 5: Datos de la prueba unitaria a los algoritmos de interpolación y evaluación.

En la tabla se puede apreciar que el valor esperado se corresponde con el valor actual para los juegos de datos utilizados, haciéndose extensible este resultado a todos los juegos de datos posibles. Por tanto queda demostrado que los algoritmos de interpolación y evaluación son precisos y fiables.

[9] Puntos a interpolar.

[10] Puntos en los que se evalúa el polinomio resultante de la interpolación.

Prueba 2: Para evaluar el funcionamiento de los algoritmos de cálculo de área, velocidad y aceleración se diseñó una prueba con doce escenarios distintos. Los valores actuales se calcularon utilizando la herramienta Euler y los valores esperados se obtuvieron mediante los algoritmos de cálculo de área, velocidad y aceleración, previa preparación de los datos mediante la interpolación de cien puntos generados en el intervalo (0;20) a partir de los escenarios en cuestión.

Funciones	Intervalo	Valores actuales	Valores esperados
100*sen(x)	(1;19)	311.283	311.283
100*sen(x)	(5;15)	173.428	173.428
20000/x*sen(2x)	(1;19)	1224.5	1224.5
20000/x*sen(2x)	(5;15)	6829.03	6829.03
x*sen(2x)/cosh(x/8)	(1;19)	76.3017	76.3017
x*sen(2x)/cosh(x/8)	(5;15)	36.7416	36.7416

Tabla 6: Datos de la prueba unitaria al algoritmo de cálculo de área.

Funciones	Posición	Valores actuales (V [11])	Valores esperados (V)	Valores actuales (A [12])	Valores esperados (A)
100*sen(x)	15	1.68586	1.68586	0.007884	0.007884
100*sen(x)	5	1.73869	1.73869	0.002654	0.002654
20000/x*sen(2x)	13	3.61081	3.61081	0.266248	0.266248
20000/x*sen(2x)	11	3.07331	3.07331	0.27119	0.27119
x*sen(2x)/cosh(x/8)	14	0.900103	0.900103	0.05345	0.05345
x*sen(2x)/cosh(x/8)	6	0.412703	0.412703	0.066732	0.066732

Tabla 7: Datos de la prueba unitaria a los algoritmos de cálculo de velocidad y aceleración.

En las tablas se puede apreciar que el valor esperado se corresponde con el valor actual

[11] Velocidad (primera derivada de la ecuación del desplazamiento en función del tiempo).

[12] Aceleración (segunda derivada de la ecuación del desplazamiento en función del tiempo).

para todos los escenarios, haciéndose extensible este resultado a todos los juegos de datos posibles. Por tanto queda demostrado que los algoritmos de cálculo de velocidad y aceleración son precisos y fiables.

3.2 Pruebas al dispositivo de entrada de datos.

La interfaz que permite captar los datos de la ejecución de un estudio a un un sujeto se realiza mediante una pantalla táctil marca MagicTouch de diecisiete pulgadas que se fija al monitor, tanto su controlador como su modo de toque están basados en el funcionamiento del ratón –mouse– con un tiempo de respuesta máximo de diez milisegundos y una precisión de toque de tres milímetros de error como máximo, con una resolución de 4096X4096 píxeles y una fuerza de activación de cincuenta a ciento veinte gramos por centímetro cuadrado (58).

Para comprobar si el proceso de calibración de la pantalla influye de manera significativa en la precisión de los toques se diseñó una prueba con cuatro escenarios distintos donde una misma persona después de calibrar la pantalla táctil debía tocar en diez ciclos para cada escenario cuatro puntos ubicados a seis milímetros aproximadamente de cada esquina y un punto central (Fig. 12) intentando ser lo más preciso posible.

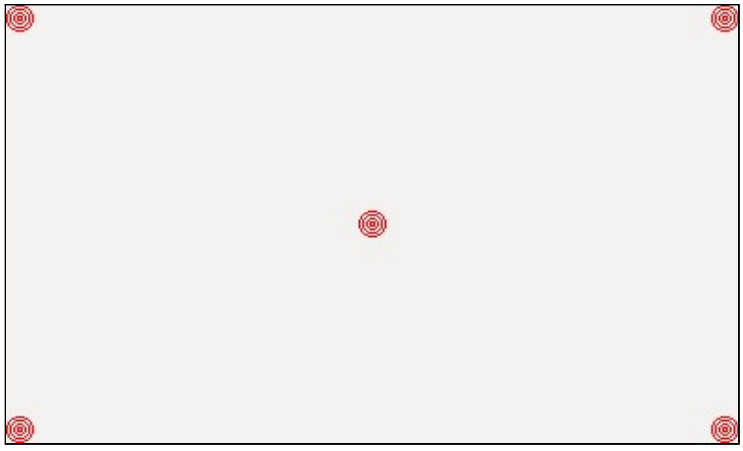

Fig. 12: Posición de los puntos para la prueba al dispositivo de entrada de datos.

Calibración	Ciclos	E[13] - Punto 1	E - Punto 2	E - Punto 3	E - Punto 4	E - Punto 5
1	1	1.6625	6.1859	7.6777	7.1726	2.6613
	2	3.7877	7.4288	5.907	3.6554	3.3249
	3	1.8587	5.4779	4.5454	6.8818	2.0781
	4	3.4273	3.1789	5.5916	5.9838	1.1755
	5	2.3694	4.0828	6.5715	3.4795	2.1192
	6	2.0781	4.6467	6.1777	4.8865	3.5873
	7	0.6571	2.4234	4.0669	6.2312	4.3048
	8	1.1755	4.5623	7.9394	6.8356	0.5878
	9	2.9389	3.2461	4.5623	7.5843	4.8201
	10	2.8944	3.4398	7.9567	4.2902	2.4234
					Media general	3.6672

Tabla 8: Datos de la primera calibración.

Calibración	Ciclos	E - Punto 1	E - Punto 2	E - Punto 3	E - Punto 4	E - Punto 5
2	1	4.5711	5.6147	7.1142	3.7636	3.2607
	2	1.664	5.5916	5.8556	3.7406	4.9579
	3	3.9538	4.7388	6.1786	6.31	3.1362
	4	0.836	3.5266	3.8365	4.1144	3.2572
	5	3.1765	5.653	6.7658	5.7963	2.6758
	6	5.8335	2.645	6.6043	5.653	4.2347
	7	3.6352	1.4694	6.0586	5.3109	1.0797
	8	2.0639	5.8556	4.4083	2.997	2.5888
	9	4.2078	5.0066	5.5902	4.1836	1.6872
	10	1.9641	6.9359	7.4972	3.9429	1.2934
					Media general	3.8157

Tabla 9: Datos de la segunda calibración.

[13] Error en milímetros de la respuesta respecto al objetivo.

Calibración	Ciclos	E - Punto 1	E - Punto 2	E - Punto 3	E - Punto 4	E - Punto 5
3	1	1.5375	3.5312	5.507	5.6303	3.7174
	2	3.6143	1.5621	4.4585	3.5911	2.0572
	3	2.0781	2.1192	6.2826	4.4116	4.0448
	4	3.2394	5.1862	5.8146	4.1747	4.594
	5	3.2693	5.2981	4.3689	6.9266	5.0557
	6	0.8312	5.7739	5.2572	5.3516	4.6907
	7	3.3249	3.1789	5.3862	3.825	4.9264
	8	2.1192	2.7725	5.3225	5.1325	4.8699
	9	3.1789	6.0586	1.4694	5.6454	5.4666
	10	1.9714	4.9961	4.1747	4.7309	2.7664
					Media general	3.7761

Tabla 10: Datos de la tercera calibración.

Calibración	Ciclos	E - Punto 1	E - Punto 2	E - Punto 3	E - Punto 4	E - Punto 5
4	1	1.5826	2.8944	4.1747	4.8077	4.3873
	2	5.0305	6.7122	5.4028	3.7984	1.7805
	3	2.3694	6.1786	4.7841	10.189	3.0355
	4	3.5533	5.7226	5.1102	6.178	3.547
	5	1.5567	4.3113	6.0523	4.1769	2.3687
	6	2.2382	3.2397	2.8821	6.6057	4.0639
	7	3.4934	3.6407	6.4521	3.5753	3.2923
	8	0.9303	4.5447	6.7644	4.8201	5.2329
	9	3.2857	4.1976	4.947	3.5958	1.4494
	10	1.7999	5.9507	4.9585	3.0844	3.9209
					Media general	3.7922

Tabla 11: Datos de la cuarta calibración.

El coeficiente de variación de los errores medios de cada calibración es de 1.74%. Para analizar si la variación entre las distintas calibraciones puede provocar alteraciones en los resultados de los exámenes, se diseñó una prueba donde intervinieron veintisiete sujetos

sanos que ejecutaron con ambas manos el examen Índice punto en su variante punto fijo, todas los exámenes fueron realizados bajo una única calibración de la pantalla. Al analizar los errores cometidos por los veintisiete sujetos en las cincuenta y cuatro ejecuciones del examen, se obtuvo un coeficiente de variación de 8.93%, comparando estos dos coeficientes de variación se demuestra que el error provocado por la calibración no afecta los resultados de los exámenes.

Basándose en los resultados de esta prueba y conociendo que el tiempo máximo de respuesta de la pantalla táctil será difícilmente igualado por el tiempo empleado por un sujeto sano para ejecutar un movimiento que es de aproximadamente trescientos milisegundos para crear un plan motor y cien milisegundos para corregir uno que haya iniciado (59), se puede concluir que el uso de la pantalla táctil de marca MagicTouch no afectará la correcta captura de los datos.

3.3 Pruebas de aceptación.

Las pruebas de aceptación fueron realizadas por el cliente para comprobar el cumplimiento de los requisitos funcionales, las facilidades de interacción, el comportamiento, el correcto funcionamiento, la captura correcta de los datos y para evaluar la interacción de los sujetos con el sistema.

Pruebas indirectas.

El software se sometió inicialmente a pruebas indirectas (sin pacientes) en el Departamento de Neurofisiología Clínica observándose un correcto funcionamiento, tiempos de respuestas rápidos, el completo cumplimiento de los requisitos funcionales, una gestión efectiva de los datos, facilidades para la adaptación de los usuarios a la nueva tecnología y adaptabilidad para crear y ejecutar nuevos estudios sobre la base de los exámenes que implementa.

Los especialistas del departamento se mostraron conformes con el diseño, la interfaz gráfica, las funcionalidades de cada uno de los módulos y la simplicidad en su empleo.

Con estos resultados se demostró que la elicitación de requisitos funcionales, el análisis y diseño, la arquitectura desarrollada y la selección de las herramientas fueron idóneos para el proceso de desarrollo del software.

Pruebas directas.

Posteriormente se efectuó una segunda etapa de pruebas con la participación de 21 pacientes enfermos con SCA2 y 40 sujetos de control, con el objetivo de evaluar la aplicación de los exámenes y observar si existía la presencia de algún comportamiento anómalo provocado por los propios trastornos de los pacientes. Además de evaluar los tiempos de ejecución y el comportamiento del software ante la sobrecarga de trabajo, la correcta captura y procesamiento de los datos, así como la interacción de los pacientes con el equipo médico.

Estas pruebas permitieron constatar que el software tiene un comportamiento estable ante la sobrecarga de trabajo, que los exámenes se aplican de forma correcta sin observarse comportamientos anómalos, que los datos obtenidos y almacenados son correctos y que la forma en que se aplican los exámenes motiva a los pacientes logrando una interacción satisfactoria.

3.4 Valoración de sostenibilidad.

Todo proyecto informático debe estar sujeto a normas de sostenibilidad en las dimensiones administrativa, socio-humanista, ambiental y tecnológica que garantizan su correcto desarrollo, despliegue, explotación y mantenimiento.

Dimensión administrativa.

Durante el desarrollo del proyecto informático el CIRAH prácticamente no incurrió en gastos, el sistema fue desarrollado por un estudiante universitario sin recibir retribución monetaria por esta actividad, utilizando una laptop personal y softwares gratuitos. Para la instalación de la solución se utilizó una computadora personal con un monitor adicional, procesador

Pentium IV, 256 MB de RAM, 120 GB de disco duro y 64 MB de memoria de vídeo, valorada en $ 725 CUC. La pantalla táctil acoplada al monitor que funciona de interfaz entre el sistema y el sujeto fue adquirida por donación.

Con la aplicación de este sistema se logra minimizar el tiempo en las actividades relacionadas con la cuantificación de la coordinación mediante las pruebas neurológicas implementadas, además de un ahorro considerable de materiales de oficina (papel, lápices, tinta, etc).

El uso del software desarrollado humaniza la aplicación de los exámenes a la vez que minimiza el margen de error en la captura y análisis de las variables neurológicas implicadas en las pruebas, contribuyendo a un aumento considerable de la calidad de los resultados.

Estas características evidencian la sostenibilidad del proyecto en la dimensión administrativa.

Dimensión socio-humanista.

El NeuroScreening contribuye con la evaluación terapéutica, la formulación de escalas clínicas objetivas, la evaluación de la rehabilitación, la descripción de las etapas evolutivas de la enfermedad, con el desarrollo de modelos pronósticos y con la búsqueda de correlaciones entre variables clínico-cuantitativas y moleculares, estos aspectos son de vital importancia en los esfuerzos para crear un protocolo de tratamiento para la SCA2 que permita mejorar de forma sustancial la calidad de vida de los afectados. El software se puede aplicar en otros centros que realicen estudios de trastornos del movimiento en los miembros superiores.

El diseño de las interfaces se realizó de forma sencilla permitiendo minimizar la curva de aprendizaje del personal que lo utiliza, además el sistema cuenta con una ayuda detallada, la que junto a las características de fiabilidad y precisión contribuyeron a la aceptación de la herramienta.

Este sistema realiza un gran aporte a la tecnología por ser una herramienta con características únicas en su campo de aplicación y por la aplicabilidad que tiene en el estudio

de los trastornos de los movimientos.

Estas características evidencian la sostenibilidad del proyecto en la dimensión socio-humanista.

Dimensión ambiental.

Aunque la aplicación del software no tiene impacto directo sobre el medio ambiente, este permite ahorrar materiales de oficina (papel, lápices, etc.), no genera contaminación por ruidos y su diseño es agradable a la vista. Los colores de las interfaces de interacción entre el sistema y el sujeto son configurables, esto permite seleccionar las combinaciones de colores adecuadas en caso que exista una afección visual que le impida al sujeto realizar las pruebas utilizando la configuración por defecto.

El tiempo de trabajo con el sistema está determinado por la activad que se realice, no obstante el tiempo de exposición no será prolongado. Las pruebas neurológicas se realizan de forma rápida, entretenida, en forma de juego sin dejar de tener en cuenta la seriedad de estas y la postura asumida por el sujeto que es objeto de estudio será de sentado correctamente. La ejecución de las pruebas no generan daño psicológico ni estrés.

La estructura del software quedó conformada por subsistemas con gran independencia, esto permite su uso por separado en la construcción de otros softwares con características similares representado un ahorro considerable de recursos y tiempo.

Estas características evidencian la sostenibilidad del proyecto en la dimensión ambiental.

Dimensión tecnológica.

El software se desarrolló teniendo en cuenta las características de la tecnología con cuenta el sistema nacional de salud pública para lograr su implantación sin mayores complicaciones, el CIRAH no escapa a estas condiciones tecnológicas por lo que fue posible implantar la solución propuesta.

Otro factor importante para la implantación de la solución fue que los especialistas destinados a utilizar el sistema tenían conocimientos básicos en informática y avanzados en los temas de cuantificación de coordinación visuomotora.

El soporte del proyecto está garantizado y se continua desarrollando nuevas versiones como parte de los convenios de colaboración entre la Universidad de Holguín y el CIRAH.

Estas características evidencian la sostenibilidad del proyecto en la dimensión tecnológica.

Conclusiones del Capítulo 3.

Al aplicar diferentes pruebas al software se puedo demostrar que los algoritmos con funcionalidades críticas en la captura y procesamiento de datos son fiables y precisos, que los datos capturados por pantalla son correctos, que el sistema es estable, que la navegabilidad por las funcionalidades es sencilla y organizada y que los estilos gráficos para aplicar los exámenes fueron correctos. Estas pruebas demuestran que el sistema desarrollado es sencillo, organizado, fiable y preciso.

La valoración de la sostenibilidad del proyecto demostró que desde la dimensión administrativa el desarrollo del sistema fue factible, sin incurrir en grandes gastos para su construcción e implantación, que desde la dimensión socio-humanista el software contribuye a las investigaciones científicas encaminadas a mejorar la calidad de vida de los afectados, que desde la dimensión ambiental contribuye al ahorro de recursos y no deteriora el medio ambiente y que desde la dimensión tecnológica es posible su explotación y mantenimiento.

Conclusiones generales.

Durante el desarrollo de esta investigación se realizó un estudio del estado del arte de la aplicación de exámenes para la neurología cuantitativa, incluyendo las principales aplicaciones informáticas destinadas a estos fines. Con este estudio se adquirieron los conocimientos necesarios en el campo de la aplicación de exámenes de neurología cuantitativa que permitieron continuar con la investigación, además se demostró la necesidad de desarrollar un sistema informático que paleara la problemática planteada.

El desarrollo de la solución informática estuvo guiado por la metodología de desarrollo AUP que permitió un entendimiento claro del negocio, actividad imprescindible para lograr una correcta elicitación de los requisitos y un diseño adecuado del sistema. La arquitectura propuesta facilitó la construcción del software, posibilitando que se elaborara por partes, de forma organizada y rápida, logrando obtener un sistema con alta cohesión y bajo acoplamiento entre los subsistemas diseñados.

En la fase de implementación se obtuvieron un conjunto de herramientas independientes que sostienen los procesos críticos del sistema, por sus características, estas herramientas pueden ser reutilizadas en otros sistemas con funcionalidades similares.

Para validar la solución informática se aplicaron un conjunto de pruebas unitarias, de captura de datos y de aceptación que arrojaron resultados satisfactorios, permitiendo afirmar que los métodos y herramientas seleccionadas para su desarrollo fueron acertados y que el sistema desarrollado es sencillo, organizado, fiable y preciso.

Se puede concluir que el objetivo planteado en esta investigación fue cumplido en su totalidad al obtener un sistema informático que permite la integración, configuración, aplicación uniforme y evaluación de los diferentes exámenes para evaluar la coordinación visuomotora en pacientes con SCA2, estas conclusiones demuestran el cumplimiento de la hipótesis planteada en la investigación.

Recomendaciones.

A partir de la experiencia adquirida en la aplicación de exámenes neurológicos durante el desarrollo del software y de la validación de los resultados de la investigación, se recomienda:

- Realizar los trámites pertinentes para validar el sistema como equipo médico con el objetivo de poder emplearlo formalmente y comercializarlo.
- Instalar el sistema en un Tablet PC, evaluar su comportamiento y la interacción entre los sujetos y la nueva interfaz para determinar si es factible el uso de esa tecnología con el objetivo de ganar en portabilidad y poder realizar estudios de campo.
- Una vez registrado como equipo médico extender su aplicación hacia otras enfermedades que presentan cuadros de trastornos del movimiento y generalizar su uso en los distintos centros de salud pública de nuestro país.
- Investigar el funcionamiento y aplicación de otros exámenes neurológicos para incluirlos en versiones futuras del sistema con el objetivo de aumentar su fortaleza y alcance en la cuantificación de otros trastornos neurológicos.

Bibliografía

1. Velázquez-Pérez L, García R, Santos F. Las ataxias hereditarias en Cuba. Aspectos históricos, epidemiológicos, clínicos, electrofisiológicos y de neurología cuantitativa. REV NEUROL. 2001;32(1):71-76.

2. Matilla A, Goold R, Giunti P. Molecular pathogenesis of spinocerebellar ataxias. Brain. 2006 May;129(6):1357-1370.

3. Velázquez L. ¿Qué es la Ataxia?::CIRAH [Internet]. ¿Qué es la ataxia?::CIRAH. 2010 Aug [cited 2011 Jan 18];Available from: http://www.ataxiacubana.sld.cu/code/ataxia.html

4. Paneque M, Lemos C, Escalona K, Prieto L, Reynaldo R, Velázquez M, et al. Psychological Follow-up of Presymptomatic Genetic Testing for Spinocerebellar Ataxia Type 2 (SCA2) in Cuba. Journal of Genetic Counseling. 2007 Aug;16(4):469-479.

5. Velázquez L, Almaguer LE, Escalona K, Rodríguez R, Paneque M, Laffita JM, et al. Molecular epidemiology of spinocerebellar ataxias in Cuba: Insights into SCA2 founder effect in Holguin. Neuroscience Letters. 2009 Apr;454(2):157-160.

6. Velázquez L, Medina EE. Evaluación neurofisiológica en pacientes afectados por ataxia espinocerebelosa tipo 2. REV NEUROL. 1998;27(160):921-926.

7. Paulson HL. The spinocerebellar ataxias. J Neuroophthalmol. 2009 Jun;23(3):227-237.

8. Velázquez-Pérez L, de la Hoz-Oliveros J, Hechavarría-Pupo R, Herrera-Domínguez H, Pérez-González RM. Análisis automático de los movimientos alternativos de miembros superiores en pacientes con ataxia espinocerebelosa tipo 2. REV NEUROL. 2001;33(1):10-16.

9. Velázquez-Pérez L, de la Hoz-Oliveros J, Pérez-González RM, Hechavarría-Pupo R, Herrera-Domímguez H. Evaluación cuantitativa de los trastornos de la coordinación en pacientes con ataxia espinocerebelosa tipo 2 cubana. REV NEUROL. 2001;32(7):601-606.

10. Clínica Cubana para la Investigación de la Ataxia, Holguín. Sistema automatizado para la cuantificación de los principales trastornos de la coordinación de los movimeintos en afecciones neurológicas. Modelo: ATAX-1. 2000 Jul;

11. Hernández-Sampieri R, Fernández-Collado C, Baptista-Lucio P. Metodología de la Investigación. 4th ed. México: McGRAW-HILL/INTERAMERICANA EDITORES, S.A. DE C.V. 2006.

12. Hernández-León RA, Coello-González S. El Paradigma Cuantitativo de la Investigación Científica. 1st ed. Ciudad de la Habana: EDUNIV, Editorial Universitaria; 2002.

13. Velázquez-Pérez L, Rodríguez-Labrada R, García-Rodríguez JC, Almaguer-Mederos LE, Cruz-Mariño T, Laffita-Mesa JM. A Comprehensive Review of Spinocerebellar Ataxia Type 2 in Cuba. Cerebellum. 2011;

14. Schöls L, Bauer P, Schmidt T, Schulte T, Riess O. Autosomal dominant cerebellar ataxias: clinical features, genetics, and pathogenesis. The Lancet Neurology. 2004 May;3(5):291-304.

15. Velázquez-Pérez L, Santos FN, García R, Paneque HM, Hechavarría PR. Epidemiología de la ataxia hereditaria cubana. REV NEUROL. 2001;32:606 - 611.

16. Díaz JCR, Pérez CLV, Cruz GS, Mederos LA, Gotay DA, Fernández JCG, et al. Evaluación de la restauración neurológica en pacientes con ataxia SCA2 cubana. Plasticidad y Restauración Neurológica. 2008;7(1-2):13-18.

17. Paneque HM, Prieto AL, Reynaldo RR, Cruz MT, Santos FN, Almaguer ML, et al. Psychological Aspects of Presymptomatic Diagnosis of Spinocerebellar Ataxia Type 2in Cuba. Community Genetics. 2007;10:132–139.

18. Milton JG. Quantitative Neuroscience: From Chalk Board to Bedside. Math. Model. Nat. Phenom. 2010;5(2):1-4.

19. DG O, R E, T P, J A, R F. Dominantly inherited livopontocerebellar atrophy from eastern Cuba. Clinical, neuropathological and biochemical findings. J Neurol Sci. 1989;93:37-50.

20. Pérez-Ávila I, Fernández-Vieitez JA, Martínez-Góngora E, Ochoa-Mastrapa R, Velázquez-Manresa MG. Efectos de un programa de ejercicios físicos sobre variablesneurológicas cuantitativas en pacientes con ataxiaespinocerebelosa tipo 2 en estadio leve. REVISTA DE NEUROLOGÍA. 2004;39(10):907-910.

21. Notermans NC, Dijk GW van, Graaf Y van der, Gijk J van, Wokke JHJ. Measuring ataxia: quantification based on the standard neurological examination. Journal of Neurology, Neurosurgery, and Psychiatry. 1994;57:22-26.

22. Gagnon C, Mathieu J, Mathieu J. Standardized Finger-Nose Test Validity for Coordination Assessment in an Ataxic Disorder. Can. J. Neurol. Sci. 2004;31:484-489.

23. Say MJ, Jones R, Scahill RI, Dumas EM, Coleman A, Santos RCD, et al. Visuomotor integration deficits precede clinical onset in Huntington's disease. Neuropsychologia. 2010;

24. Montcel ST du, Charles P, Ribai P, Goizet C, Bayon AL, Labauge P, et al. Composite

cerebellar functional severity score: validation of a cuantitative score of cerebellar imoairment. Brain. 2008;131:1352-1361.

25. Saunders-Pullman R, Derby C, Stanley K, Floyd A, Bressman S, Lipton RB, et al. Validity of spiral analysis in early Parkinson's disease. Mov Disord. 2008 Mar;23(4):531-537.

26. Homann CN, Suppan K, Wenzel K, Giovannoni G, Ivanic G, Horner S, et al. The bradykinesia akinesia incoordination test (BRAIN TEST), an objective and user-friendly means to evaluate patients with Parkinsonism. Mov. Disord. 2000 Jul;15(4):641-647.

27. Giovannoni G, Schalkwyk J van, Fritz VU, Lees AJ. Bradykinesia akinesia inco-ordination test (BRAIN TEST): an objetive computerised assessment of upper limb motor function. J Neurol Neurosurg Psychiatry. 1999;67:624-629.

28. Pullman SL. Spiral Analysis: A New Technique for Measuring Tremor With a Digitizing Tablet. Mov. Disord. 1998 Oct;13(S3):85-89.

29. Pressman RS. Ingeniería del Software. Un enfoque práctico. Quinta Edición. McGraw-Hill; 2002.

30. CMS. Selecting a Development Approach. CENTERS for MEDICARE & MEDICAID SERVICES. 2008 Mar 27;

31. Jacobson I, Booch G, Rumbaugh J. Unified Software Development Process. Addison-Wesley; 1999.

32. Rational-Software. Rational Unified Process Best Practices for Software Development Teams. Rational Software White Paper. 2001 Nov;

33. Meloche T. The Rational Unified Process® (RUP): A Well Documented, Complete yet Complex Methodology. The Menlo Institute LLC. 2002;

34. Escribano GF. Introducción a Extreme Programming. Departamento de Sistemas Informáticos. Universidad de Castilla-La Mancha: 2002.

35. Beck K, Andres C. Extreme Programming Explained: Embrace Change. 2nd ed. Addison-Wesley Professional; 2004.

36. Solís MC. Una explicación de la programación extrema (XP). Madrid, España: 2003.

37. Ambler SW. The Agile Unified Process (AUP) Home Page [Internet]. 2006 May [cited 2011 Apr 28];Available from: http://www.ambysoft.com/unifiedprocess/agileUP.html

38. Flores E. Metodologías Ágiles. Proceso Unificado Ágil (AUP). Universidad Unión Bolivariana. 2009;

39. Lenguaje de programación. Enciclopedia Compacta Británica. 2011;

40. Glosario de términos informáticos [Internet]. 2011 Apr 27 [cited 2011 Apr 27];Available from: http://usuarios.multimania.es/hv1102/glosario.html#lenguajedeprogramacion

41. Deitel HM, Deitel PJ, Nieto TR, Yaeger CH, Zlatkina M, Listfield J. C# How to Program. Second Edition. Deitel & Associates, Inc. 2002.

42. Budd T. Introducción a la Programación Orientada a Objetos. Buenos Aires, Argentina: Addison-Wesley; 1994.

43. Jalón JG de, Rodríguez JI, Sarriegui JM, Brazález A. Aprenda C++ como si estuviera en primero. UNIVERSIDAD DE NAVARRA: 1998.

44. Glassborow F. A Beginner's Introduction to Computer Programming. UK: John Wiley & Sons Ltd; 2004.

45. Ferguson J, Patterson B, Beres J, Boutquin P, Gupta M. La Biblia de C#. España: ANAYA; 2003.

46. Pozo P, Ponte L. Información para programadores de Microsoft .NET Framework [Internet]. CLIKEAR.COM. [cited 2011 May 11];Available from: http://www.clikear.com/manuales/csharp/index.asp

47. Nagel C, Evjen B, Glynn J, Skinner M, Watson K. Professional C# 2008. Wiley Publishing, Inc. 2008.

48. Holzner S. La Biblia de Java 2. Anaya Multimedia; 2005.

49. Programación visual de aplicaciones con C++ Builder [Internet]. [cited 2011 May 30];Available from: http://elvex.ugr.es/decsai/builder/

50. Bleivik KG. C++ Builder 2010 Professional Getting started [Internet]. 2010;Available from: www.oopschool.com/books/CPB2010.pdf

51. Introducing Qt Creator [Internet]. 2011 May 13 [cited 2011 May 13];Available from: http://doc.qt.nokia.com/qtcreator-2.1/creator-overview.html

52. Qt Creator IDE and tools — Qt - A cross-platform application and UI framework [Internet]. 2011 May 13 [cited 2011 May 13];Available from: http://qt.nokia.com/products/developer-tools

53. Galassi M, Davies J, Theiler J, Gough B, Jungman G, Alken P, et al. GNU Scientific Library. 2010;

54. Gómez MM. IMPLEMENTACIÓN DE MOVIMIENTOS SUAVES EN ROBOTS INDUSTRIALES MEDIANTE SPLINES CÚBICOS. APLICACIÓN A UN LABORATORIO REMOTO DE ROBÓTICA. 2010;

55. CORDERO JM. PROYECTO CÁLCULO INTEGRAL. APLICACIONES DE LA INTEGRAL DEFINIDA. 2005;

56. Herrera AC, Patritti H. Aplicaciones de las derivadas. Montevideo, Uruguay: Biblioteca Nacional; 2004.

57. Welcome to Euler for GTK+. 2005;

58. KEYTEC add-on magic touch screen for 16-17" LCD and CRT monitors. [Internet]. [cited 2011 Jun 27];Available from: http://www.magictouch.com/KTMT-1700.html

59. Desmurget M, Epstein CM, Turner RS, Prablanc C, Alexander GE, Grafton ST. Role of the posterior parietal cortex in updating reaching movements to a visual target. Nature Neuroscience. 1999 Jun;2(6).

Anexo 1: Descripción de los Casos de uso en formato de alto nivel.

Caso de Uso	Gestionar sujeto.
Actores	Técnico.
Importancia	Alto
Descripción	El técnico registra, visualiza o actualiza un sujeto.
Referencia	RF-1, RF-2 y RF-3.

Tabla 12: Caso de uso Gestionar sujeto.

Caso de Uso	Gestionar estudio.
Actores	Técnico.
Importancia	Alto
Descripción	El técnico crea, visualiza o actualiza un estudio.
Referencia	RF-4, RF-5 y RF-6.

Tabla 13: Caso de uso Gestionar estudio.

Caso de Uso	Gestionar prueba.
Actores	Técnico.
Importancia	Alto
Descripción	El técnico crea, visualiza, actualiza o elimina una prueba.
Referencia	RF-7, RF-8, RF-9, RF-10 y Caso de uso Administrar funciones matemáticas (extendido).

Tabla 14: Caso de uso Gestionar prueba.

Caso de Uso	Administrar funciones matemáticas.
Actores	-
Importancia	Medio
Descripción	El técnico crea, modifica o visualiza una función matemática con el objetivo de gestionar una prueba de tipo Persecución motora.
Referencia	RF-11.

Tabla 15: Caso de uso Administrar funciones matemáticas.

Caso de Uso	Ejecutar estudio
Actores	Técnico (inicia) y Sujeto.
Importancia	Alto
Descripción	El técnico aplica un estudio a un sujeto, el sujeto completa todas las pruebas impuestas y al concluir, si se desarrolló de forma correcta, el técnico guarda los resultados del estudio, si no, el técnico repite el estudio al sujeto.
Referencia	RF-12 y RF-13.

Tabla 16: Caso de uso Ejecutar estudio.

Caso de Uso	Buscar sujetos.
Actores	Técnico.
Importancia	Bajo
Descripción	El técnico realiza búsquedas de sujetos por diferentes directorios, de forma simple o compuesta.
Referencia	RF-14 y Caso de uso Filtrar sujetos (extendido).

Tabla 17: Caso de uso Gestionar directorios con ficheros de sujetos.

Caso de Uso	Filtrar sujetos
Actores	Técnico.
Importancia	Bajo
Descripción	El técnico filtra sujetos por diferentes criterios.
Referencia	RF-15.

Tabla 18: Caso de uso Filtrar sujetos.

Caso de Uso	Generar reportes.
Actores	Técnico.
Importancia	Medio
Descripción	El técnico genera reportes de los resultados de los estudios aplicados a uno o varios sujetos. Los reportes se pueden generar por estudios o por sujetos y pueden ser simples, generales o comparativos. Los reportes se pueden imprimir o ser exportados a diferentes formatos.
Referencia	RF-16, RF-17, RF-18, RF-19, RF-20, RF-21, RF-22, RF-23, RF-24 y RF-25.

Tabla 19: Caso de uso Generar reportes.

Caso de Uso	Configurar sistema
Actores	Técnico.
Importancia	Bajo
Descripción	El técnico configura diferentes parámetros del sistema y de la ejecución de las pruebas.
Referencia	RF-26 y RF-27.

Tabla 20: Caso de uso Configurar sistema.

Anexo 2: Diagrama de clases del sistema.

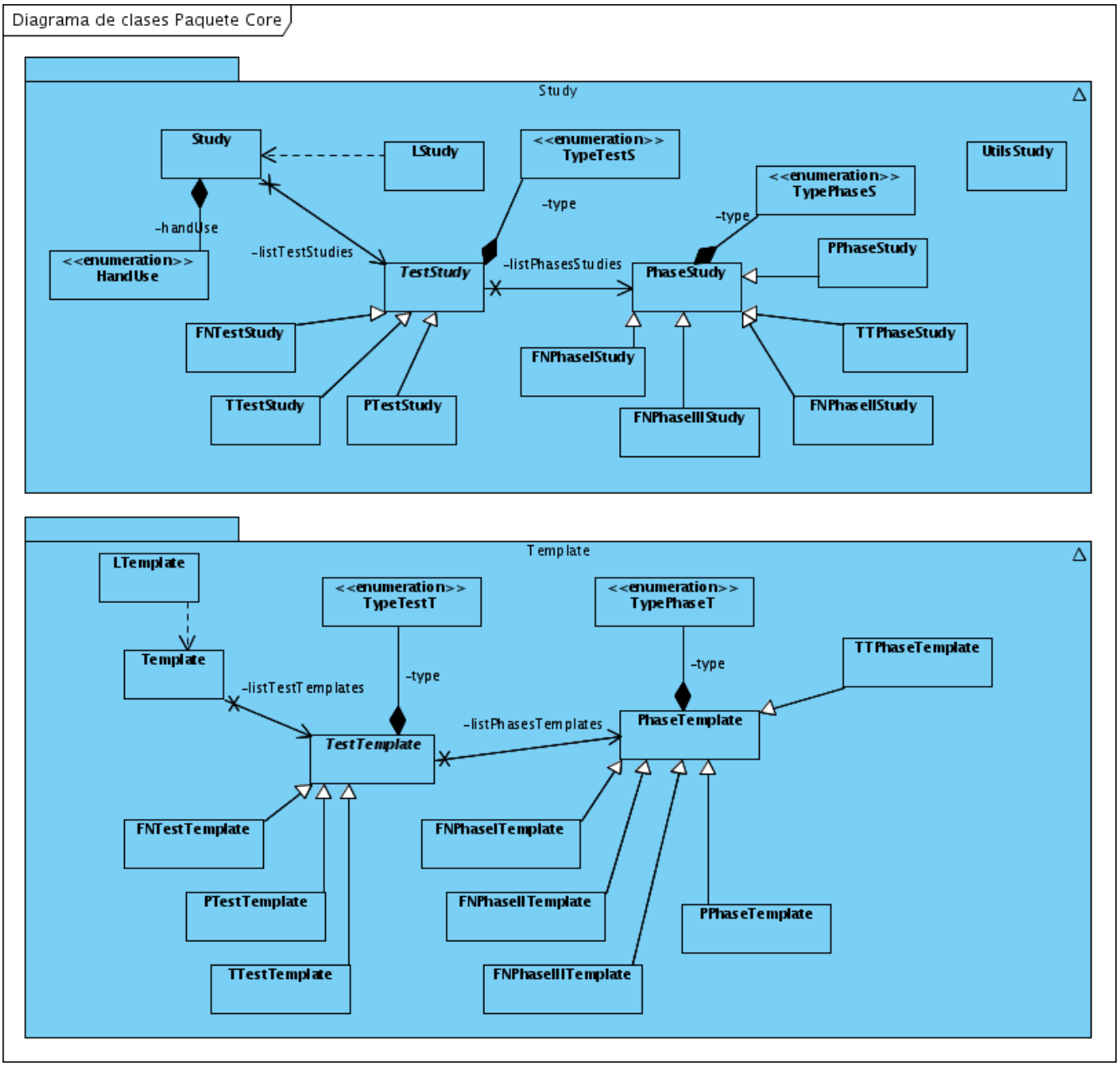

Fig. 13: Diagrama de clases del sistema, paquete CORE (parte 1).

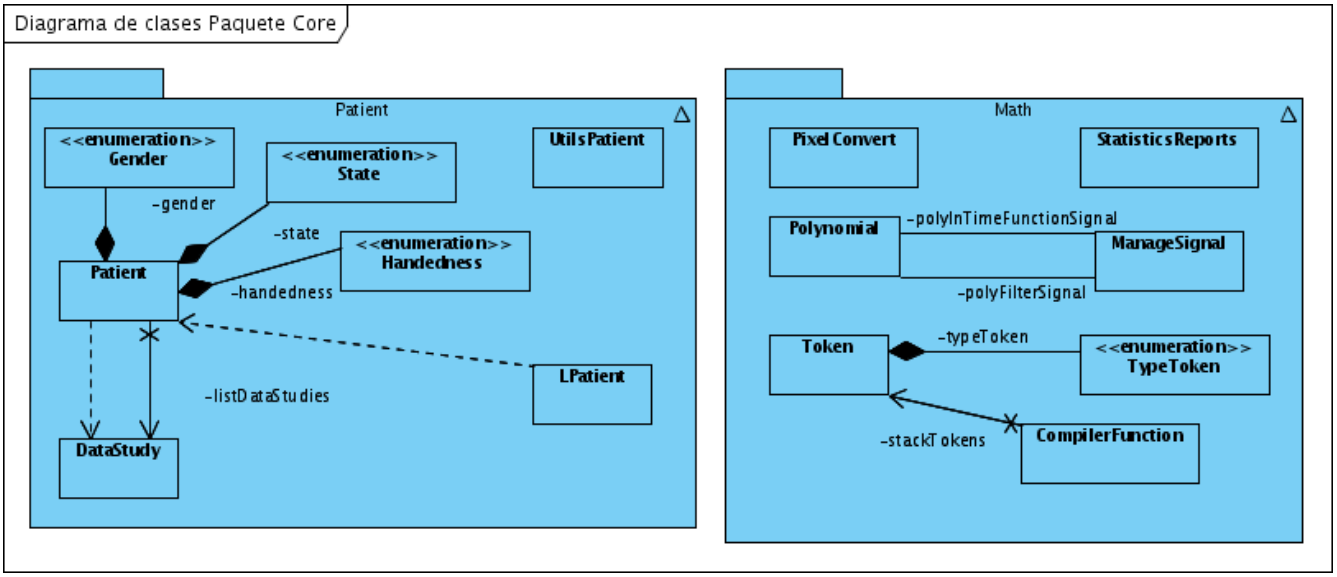

Fig. 14: Diagrama de clases del sistema, paquete CORE (parte 2).

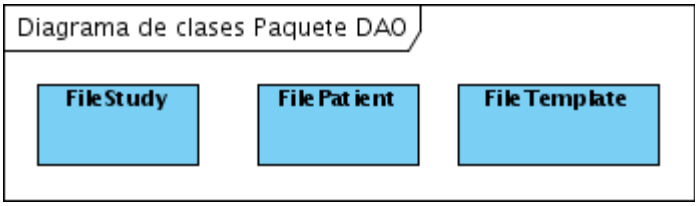

Fig. 15: Diagrama de clases del sistema, paquete DAO.

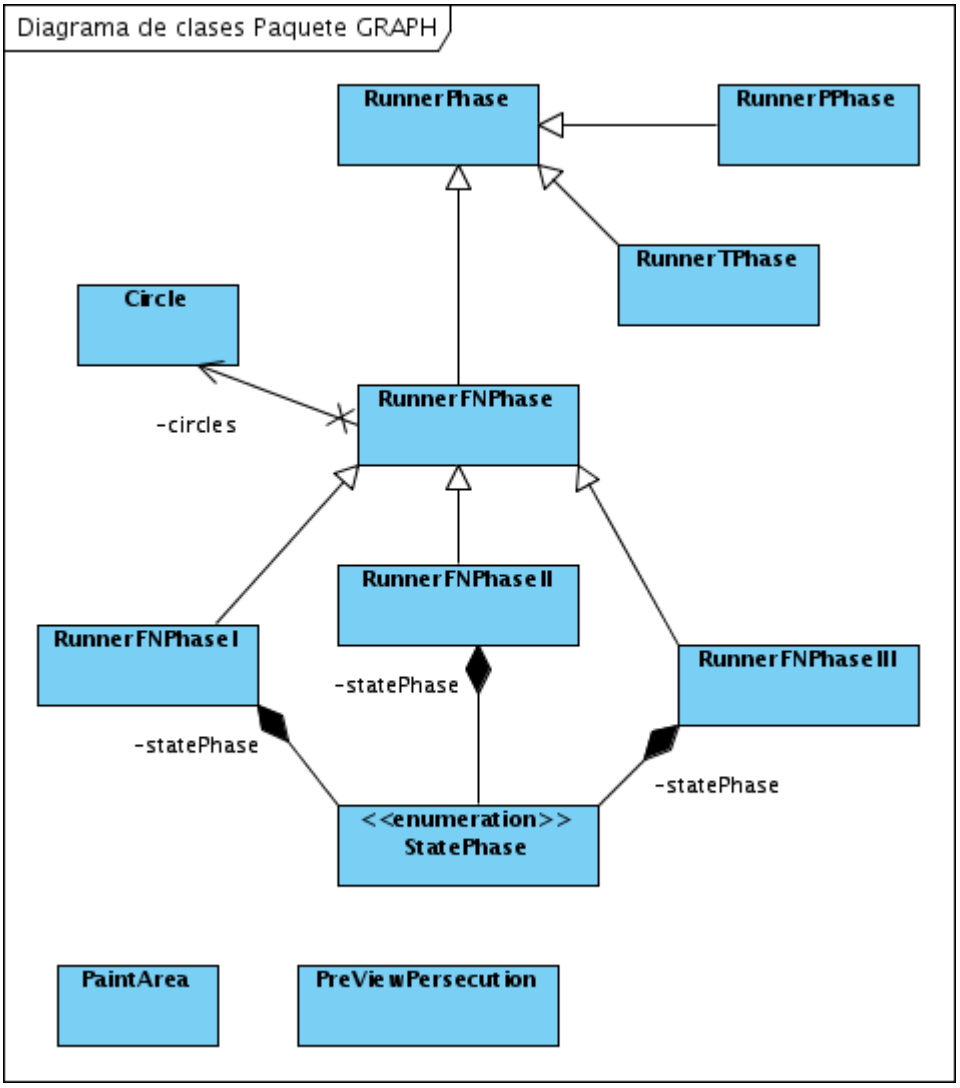

Fig. 16: Diagrama de clases del sistema, paquete GRAPH.

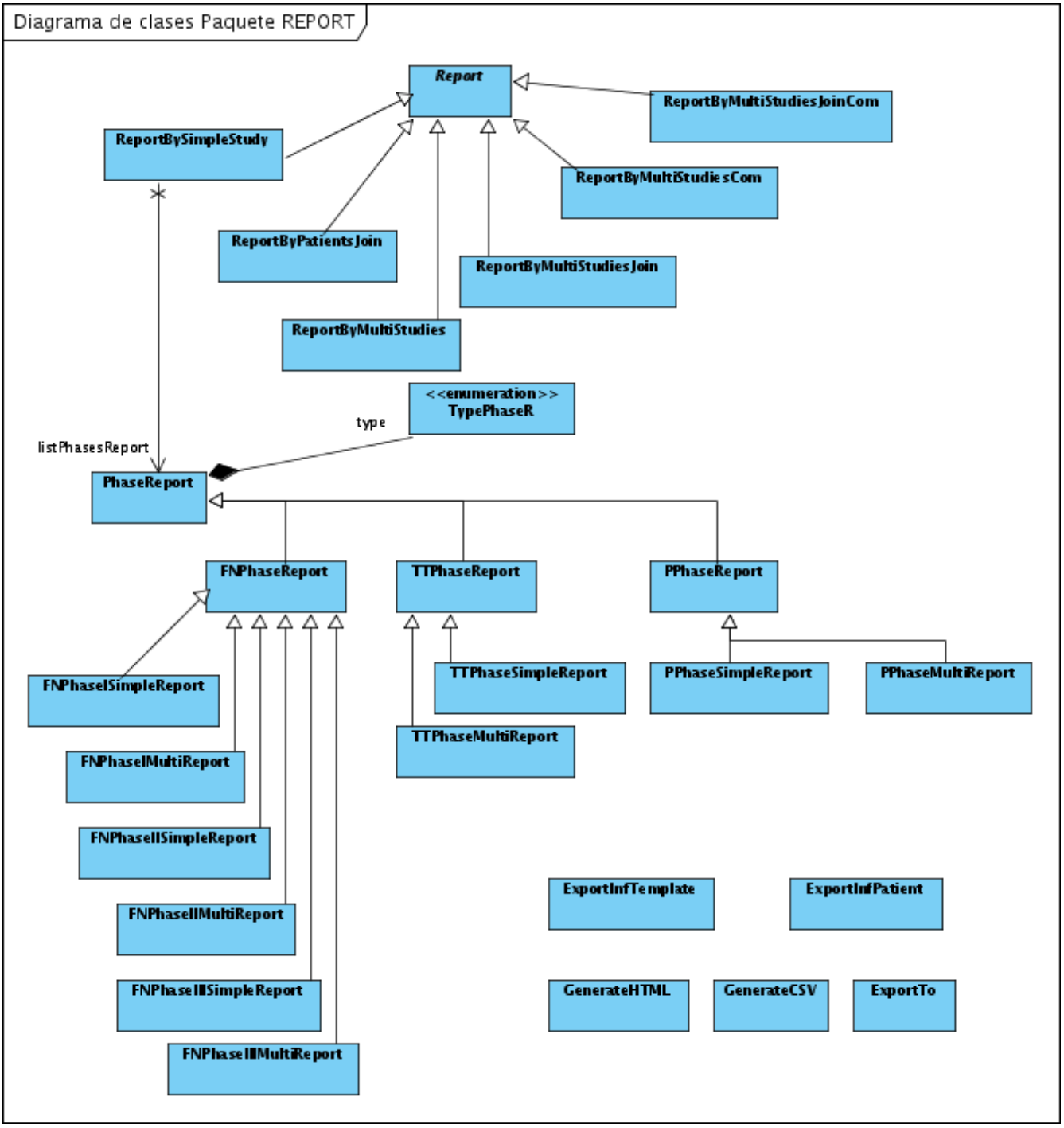

Fig. 17: Diagrama de clases del sistema, paquete REPORT.

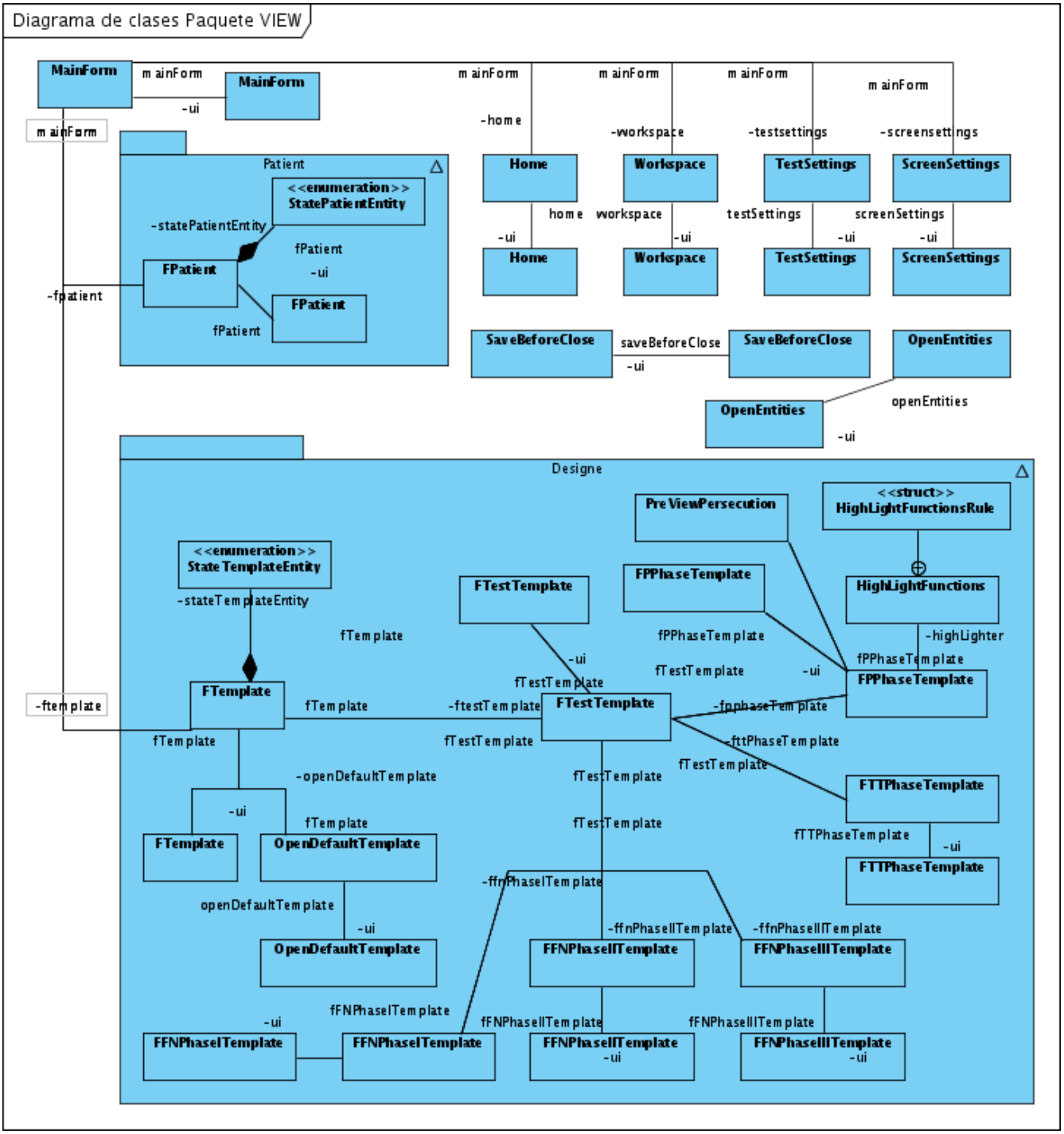

Fig. 18: Diagrama de clases, paquete VEW (parte 1).

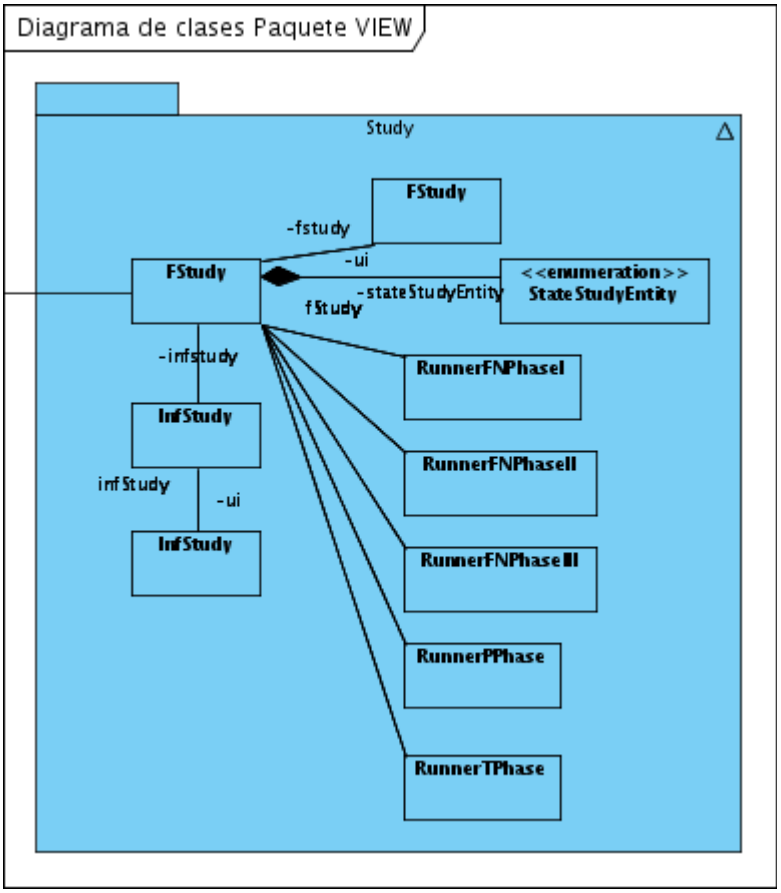

Fig. 19: Diagrama de clases, paquete VIEW (parte 2).

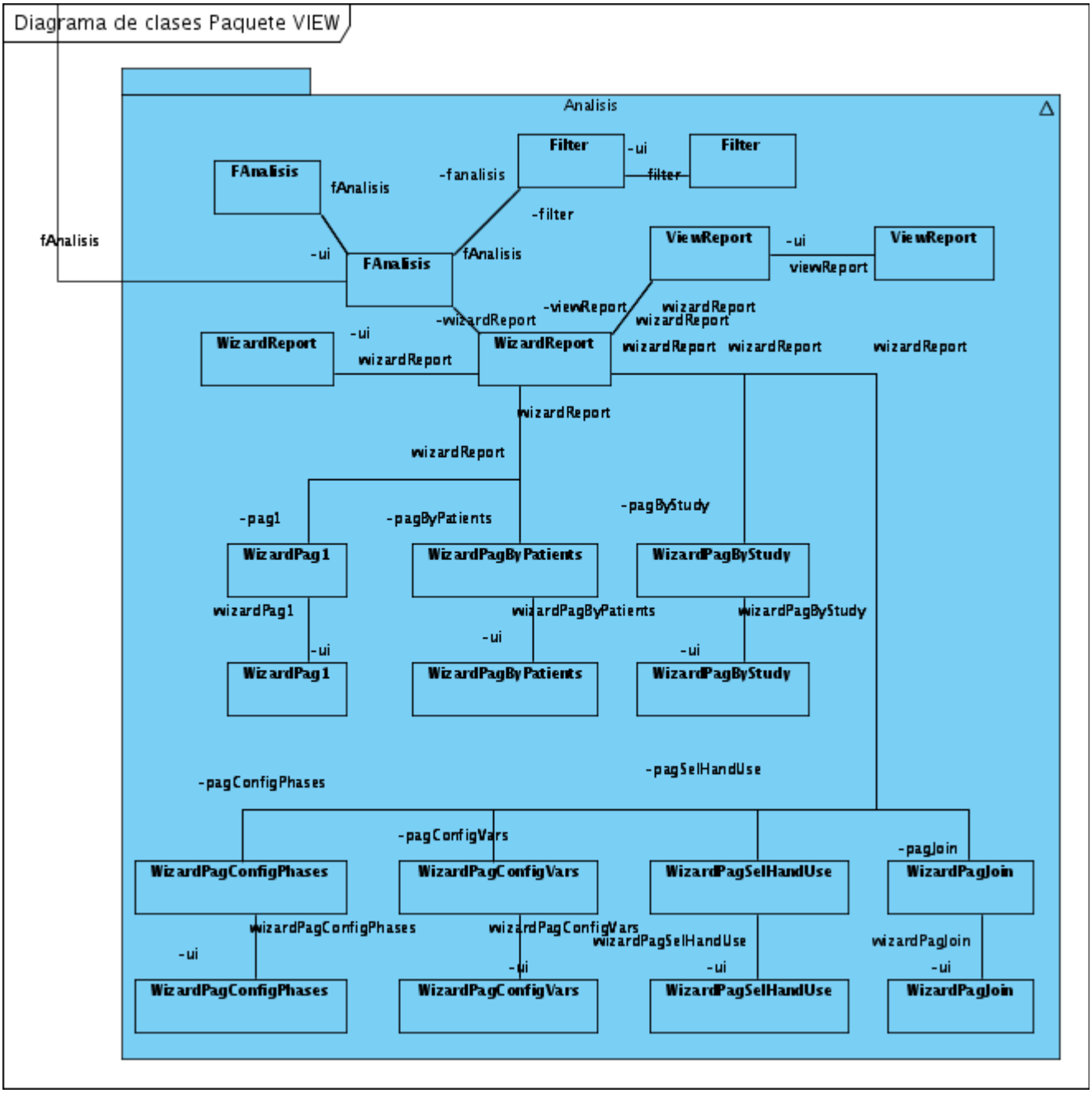

Fig. 20: Diagrama de clases, paquete VIEW (parte 3).

Anexo 3: Diagrama de clases del Intérprete de funciones matemáticas.

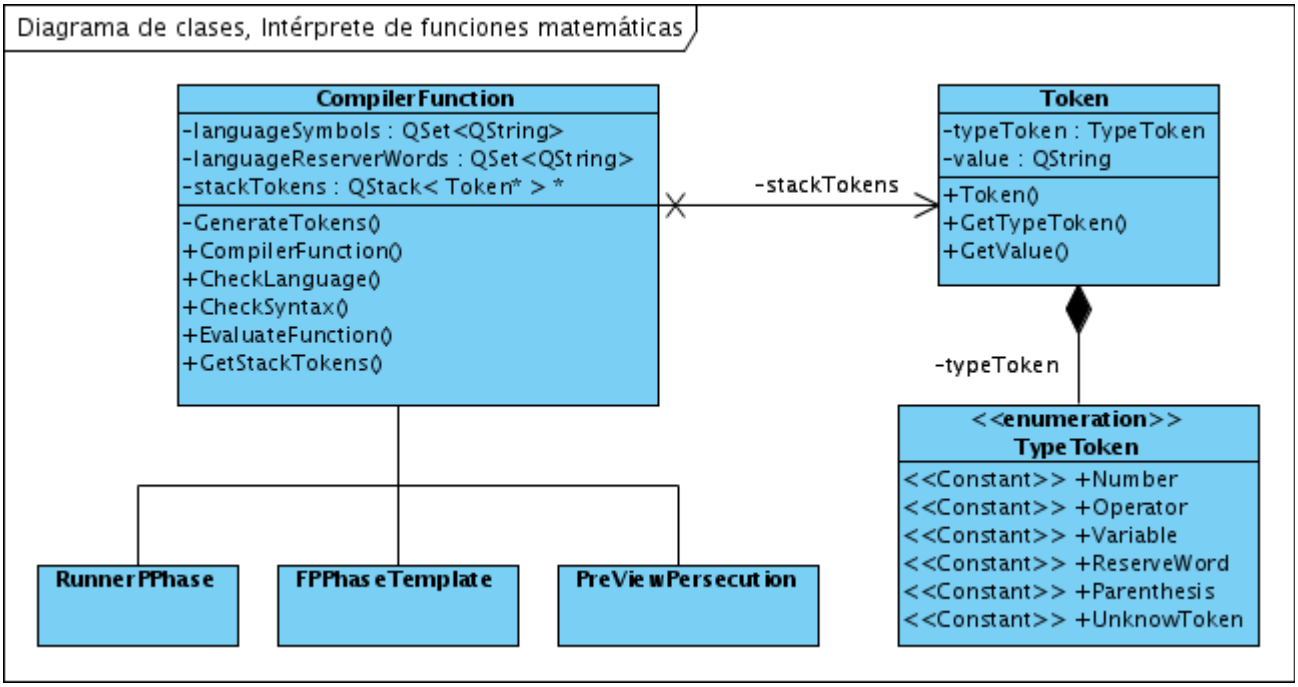

Fig. 21: Diagrama de clases del Intérprete de funciones matemáticas.

Las clases CompilerFunction, Token y TypeToken pertenecen al intérprete, las clases PreViewPersecution, RunnerPPhase y FPPhaseTemplate consumen las funcionalidades de este.

Anexo 4: Diagrama de clases del Procesador de señales de movimiento.

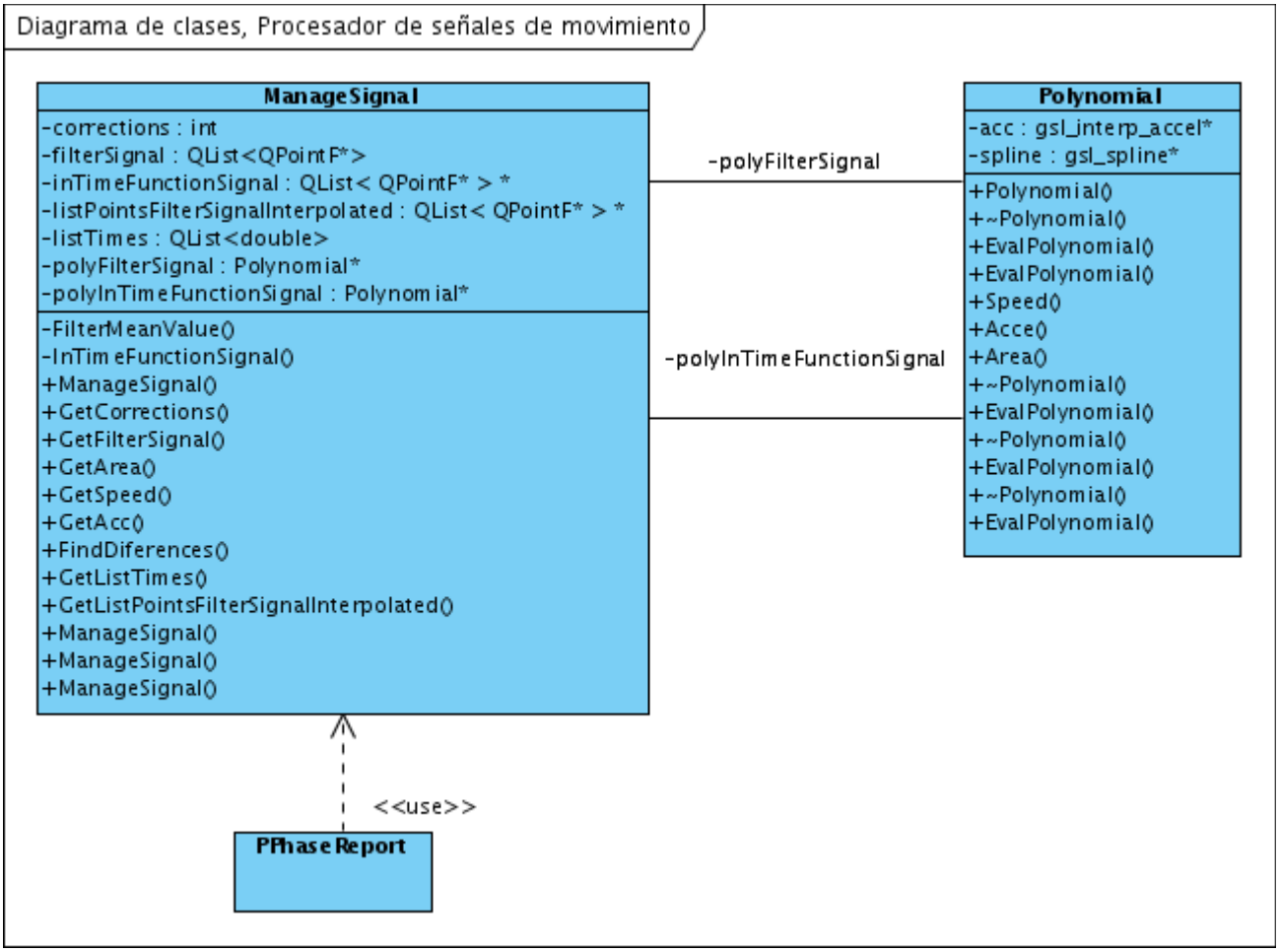

Fig. 22: Diagrama de clases del Procesador de señales de movimiento.

Las clases ManageSignal y Polynomial pertecen al Procesador, la clase PPhaseReport consume las funcionalidades de este.